

BAYESUVIUS

A VISUAL DICTIONARY OF BAYESIAN
NETWORKS AND CAUSAL INFERENCE



ROBERT R. TUCCI

Bayesuvius,
a visual dictionary of Bayesian Networks and
Causal Inference

Robert R. Tucci
www.ar-tiste.xyz

May 17, 2022

This book is constantly being expanded and improved. To download
the latest version, go to <https://github.com/rrtucci/Bayesuvius>

Bayesuvius

by Robert R. Tucci

Copyright ©2020-2021, Robert R. Tucci.

This work is licensed under the Creative Commons Attribution-Noncommercial-No Derivative Works 3.0 United States License. To view a copy of this license, visit the link <https://creativecommons.org/licenses/by-nc-nd/3.0/> or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042.



Figure 1: View of Mount Vesuvius from Pompeii

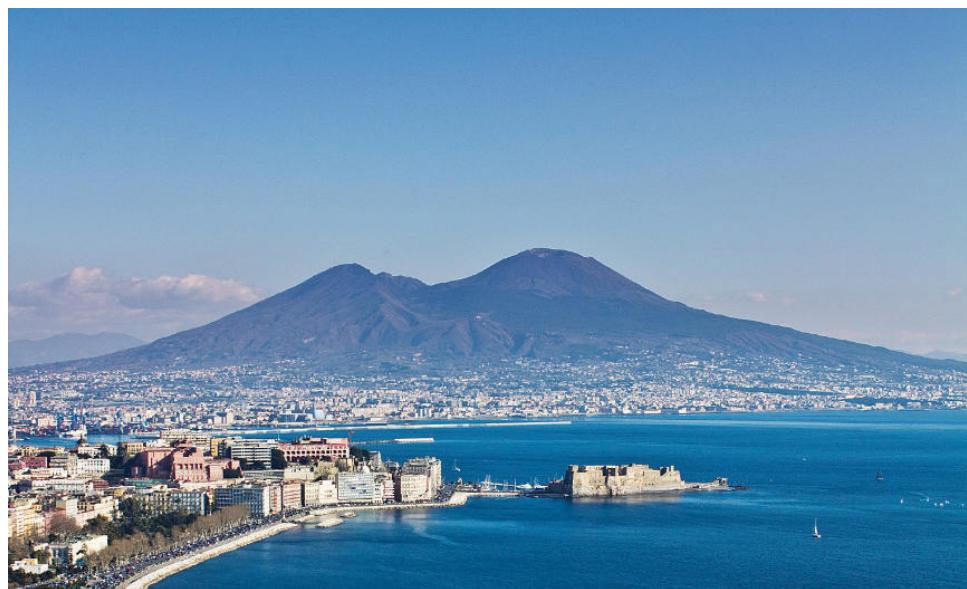


Figure 2: Mount Vesuvius and Bay of Naples

Contents

Foreword	14
Navigating the ocean of Judea Pearl's Books	15
CI-2-3 track	16
Notational Conventions and Preliminaries	19
0.1 Some abbreviations frequently used throughout this book	19
0.2 $\mathcal{N}(!a)$	19
0.3 One hot vector	19
0.4 Special sets	20
0.5 Kronecker delta function	20
0.6 Dirac delta function	20
0.7 Indicator function (aka Truth function)	20
0.8 Majority function	20
0.9 Underlined letters indicate random variables	21
0.10 Probability distributions	21
0.11 Discretization of continuous probability distributions	21
0.12 Samples, i.i.d. variables	22
0.13 Expected Value and Variance	23
0.14 Conditional Expected Value	23
0.15 Notation for covariances	23
0.16 Conditional Covariance	24
0.17 Law of Total Variance	24
0.18 Normal Distribution	25
0.19 Uniform Distribution	27
0.20 Softmax function (aka Boltzmann Distribution)	27
0.21 Sigmoid and log-odds functions	28
0.22 Estimand, Estimator, Bias	29
0.23 Maximum Likelihood Estimate, Likelihood Ratio Test	29
0.24 Mean Square Error (MSE)	30
0.25 Bayes Rule, Bayesian Updating And Conjugate Priors	33
0.26 Linear regression, Ordinary Least Squares (OLS)	34

0.26.1	LR, assuming x_σ are non-random	35
	Derivation of LR From Minimization of Error	35
	Geometry of LR with non-random x_σ	36
	LR Goodness of Fit, R^2	37
0.26.2	LR, assuming x_σ are random	39
	Transforming expressions from non-random to random x_σ	40
	Geometry of LR with random x_σ	42
	Regression interpreted as differentiation of y	43
0.27	Logistic Regression (LoR)	44
0.28	Entropy, Kullback-Liebler divergence, Cross-Entropy	45
0.29	Definition of various entropies used in Shannon Information Theory	46
0.30	Arc Strength (Arc Force)	47
0.31	Pearson Chi-Squared Test	47
0.32	Demystifying Population and Sample Variances	48
0.33	Independence of $\hat{\mu}$ and $\hat{\sigma}^2$	50
0.34	Chi-square distribution	51
0.35	Student's t-distribution	52
0.36	Hypothesis testing and 3 classic test statistics (Likelihood, Score, Wald)	55
0.37	Error Bars	58
0.38	Confidence Intervals	59
0.39	p-value, general definition	61
0.40	Short Summary of Boolean Algebra	63
Definition of a Bayesian Network		65
Bayesian Networks, Causality and the Passage of Time		68
0.41	Unifying Principle of this book	68
0.42	You say tomato, I say tomato	68
0.43	A dataset is causal model free	69
0.44	What is causality?	70
0.45	Bayesian Networks and the passage of time	71
0.46	Advice for the DAG-phobic	72
1 AdaBoost		73
1.1	AdaBoost for general ensemble of w-classifiers	73
1.2	AdaBoost for ensemble of tree stumps	77
2 ARACNE structure learning		79
3 Backdoor Adjustment Formula		81
3.1	Examples	82

4 Back Propagation (Automatic Differentiation)	86
4.1 General Theory	86
4.1.1 Jacobians	86
4.1.2 Bnets for function composition, forward propagation and back propagation	87
4.2 Application to Neural Networks	88
4.2.1 Absorbing b_i^λ into w_{ij}	88
4.2.2 Bnets for function composition, forward propagation and back propagation for NN	90
4.3 General bnets instead of Markov chains induced by layered structure of NNs	93
5 Basic Curve Fitting Using Gradient Descent	94
6 Bell and Clauser-Horne Inequalities in Quantum Mechanics	96
7 Berkson's Paradox	97
8 Binary Decision Diagrams	99
9 Chow-Liu Trees and Tree Augmented Naive Bayes (TAN)	103
9.1 Chow-Liu Trees	103
9.2 Tree Augmented Naive Bayes (TAN)	107
10 Counterfactual Reasoning	109
10.1 The 3 Rungs of Causal AI	109
10.2 Do operator	110
10.3 Imagine operator	110
11 Cross-Validation	113
12 Dataset Shift and Batch Normalization	116
12.1 Covariate Shift	117
12.2 Concept Shift	117
12.3 Batch Normalization	118
13 Decision Trees	119
13.1 Transforming a dtree into a bnet	121
13.2 Structure Learning for Dtrees	122
13.2.1 Information Gain, Gini	123
13.3 Information Gain Ratio	126
13.3.1 Pseudo-code	126
14 Decisions Based on Rungs 2 and 3: COMING SOON	129

15 Difference-in-Differences	130
15.1 John Snow, DID and a cholera transmission pathway	130
15.2 PO analysis	132
15.3 Linear Regression	134
16 Digital Circuits	137
16.1 Mapping any dcircuit to a bnet	137
16.1.1 Option A of Fig.16.2	137
16.1.2 Option B of Fig.16.2	138
17 Do Calculus	139
17.1 3 Rules of Do Calculus	142
17.2 Parent Adjustment Formula	143
17.3 Backdoor Adjustment Formula	145
17.4 Frontdoor Adjustment Formula	146
17.5 Comparison of Backdoor and Frontdoor adjustment formulae	147
17.6 Do operator for DEN diagrams	148
18 Do Calculus proofs	151
19 D-Separation	168
20 D-Separation and Do Calculus Rules for LDEN: COMING SOON	171
21 D-Separation in Quantum Mechanics	172
22 Dynamical Bayesian Networks	173
23 Expectation Maximization	175
23.1 The EM algorithm:	176
23.1.1 Motivation	177
23.2 Minorize-Maximize (MM) algorithms	177
23.3 Examples	179
23.3.1 Gaussian mixture	179
23.3.2 Blood Genotypes and Phenotypes	180
23.3.3 Missing Data/Imputation	182
24 Frisch-Waugh-Lovell (FWL) theorem	183
24.1 FWL, assuming x^σ are non-random	183
24.2 FWL, assuming x^σ are random	184
25 Frontdoor Adjustment Formula	187
25.1 Examples	188

26 Gaussian Nodes with Linear Dependence on Parents	189
27 Generative Adversarial Networks (GANs)	192
28 Goodness of Causal Fit	197
28.1 Abstract	197
28.2 Introduction	197
28.3 Goodness of Statistical Fit	198
28.4 GCF example 1	199
28.5 GCF example 2	201
28.6 GCF in general	202
29 Granger Causality	204
30 Hidden Markov Model	207
30.1 Calculating $P(x_t, v^n)$ and $P(x_t, x_{t+1}, v^n)$	209
30.2 Calculating \mathcal{F}_t and $\bar{\mathcal{F}}_t$	210
30.3 Calculating $P(x^n v^n)$	211
30.4 Calculating $P(v^n A, B, \pi)$	212
30.5 Calculating \hat{x}^n (Viterbi algorithm)	213
30.6 Calculating \hat{A} , \hat{B} , $\hat{\pi}$ (Baum-Welch algorithm)	214
31 Influence Diagrams & Utility Nodes	217
32 Instrumental Inequality and beyond	219
32.1 I-inequality	219
32.1.1 I-inequality for binary z,d,y	221
32.2 Bounds on Effect of IV on treatment outcome y	222
33 Instrumental Variables	225
33.1 δ with unmeasured confounder	225
33.2 δ (with unmeasured confounder) can be inferred via IV	226
33.3 More general bnets with IVs	227
33.4 Instrumental Inequality	228
34 Jackknife Resampling	229
34.1 Case $A = A^n(\vec{x}) = \frac{1}{n} \sum_{\sigma} x^{\sigma}$	231
35 Junction Tree Algorithm	233
36 Kalman Filter	234
36.1 Prediction Problem	235
36.2 Solution	236
36.3 Simple Example	237

36.4 Invariants	238
36.5 Derivation of Solution	238
37 Linear and Logistic Regression	240
37.1 Generalization to x with multiple components (features)	242
37.2 Alternative $V(b, m)$ for logistic regression	242
38 Linear Deterministic Bnets with External Noise	244
38.1 Example of LDEN diagram	244
38.2 Fully Connected LDEN diagrams	245
38.2.1 Fully connected LDEN diagram with $nx = 2$	246
38.2.2 Fully connected LDEN diagram with $nx = 3$	246
38.2.3 Fully connected LDEN diagram with arbitrary nx	247
38.3 Non-linear DEN diagrams	249
39 Markov Blankets	250
40 Markov Chain Monte Carlo (MCMC)	252
40.1 Inverse Cumulative Sampling	252
40.2 Rejection Sampling	254
40.3 Metropolis-Hastings Sampling	255
40.4 Gibbs Sampling	258
40.5 Importance Sampling	259
41 Markov Chains	261
42 Mediation Analysis	262
43 Message Passing (Belief Propagation)	268
43.1 Distributed Soldier Counting	268
43.2 Spring Systems	270
43.3 BP for Markov Chains	270
43.4 BP Algorithm for Polytrees	277
43.4.1 How BP algo for polytrees reduces to the BP algo for Markov chains	281
43.5 Derivation of BP Algorithm for Polytrees	282
43.6 Example of BP algo for a Tree	285
43.7 Bipartite bnets	289
43.8 BP for bipartite bnets (BP-BB)	292
43.8.1 BP-BB and general BP agree on Markov chains	293
43.8.2 BP-BB and general BP agree on tree bnets.	295
43.9 BP-BB and sum-product decomposition	297
44 Message Passing (Belief Propagation) in Quantum Mechanics	298

45 Meta-learners for estimating ATE	299
46 Missing Data, Imputation	303
46.1 Imputation via EM	304
46.2 Imputation via MCMC	307
46.3 Multiple Imputations	308
47 Monty Hall Problem	309
48 Multi-armed Bandits	311
48.1 Bnet for MAB	312
48.2 Reward functions	314
48.3 Regret functions	316
48.4 Strategies with random exploration	317
48.4.1 ϵ -greedy algorithm	317
48.4.2 ϵ_t -greedy algorithm	318
48.5 Strategies with nonrandom exploration	318
48.5.1 Upper Confidence Bounds (UCB) algorithms	318
Frequentist UCB (UCB1) algorithm	318
Bayesian UCB algorithm	319
48.5.2 Thompson Sampling MAB (TS-MAB) algorithm	320
Bnet for general TS-MAB algorithm	320
TS-MAB algorithm with Beta agent and Bernoulli environment	322
TS-MAB algorithm, skeletal reprise	323
48.5.3 Grad-MAB algorithm	324
49 Naive Bayes	327
50 Neural Networks	328
50.1 Activation Functions $\mathcal{A}_i^\lambda : \mathbb{R} \rightarrow \mathbb{R}$	329
50.2 Weight optimization via supervised training and gradient descent . .	330
50.3 Non-dense layers	332
50.4 Autoencoder NN	334
51 Noisy-OR gate	335
51.1 3 ways to interpret the parameters π_i	336
52 Non-negative Matrix Factorization	339
52.1 Bnet interpretation	339
52.2 Simplest recursive algorithm	340
53 Observationally Equivalent DAGs	341
53.1 Examples	341

54 Personalized Treatment Effects	344
54.1 Goal, Strategy and Rationale of PTE theory	345
54.2 Bnets for PTE theory	347
54.3 $ATE = PNS - AMM$	348
54.4 Probabilities Relevant to PTE theory	349
54.5 Symmetry	353
54.6 Linear Programming Problem	354
54.7 Special constraints	355
54.8 Matrix representation of probabilities	358
54.9 Bounds on Exp. Probs. imposed by Obs. Probs.	361
54.10 Bounds on PNS_3 for unspecified bnet	363
54.11 Bounds on PNS_3 for specific bnet families	369
54.12 Numerical Examples	373
55 Personalized Expected Utility	374
55.1 Goal of PEU Theory	375
55.2 Bnets for PEU Theory	376
55.3 Bounds on EU for unspecified bnet	376
55.4 Bounds on EU for specific bnet families	379
56 Potential Outcomes and Beyond	382
56.1 G and G_{den} bnets, the starting point bnets	383
56.2 G bnet with nodes $y^\sigma(0), y^\sigma(1)$ added to it.	385
56.3 Expected Values of treatment outcome y^σ	387
56.4 Translation Dictionary	387
56.5 $\mathcal{Y}_{ d,x} = \mathcal{Y}_{d d,x}$ (SUTVA)	388
56.6 Conditional Independence Assumption (CIA)	389
56.7 Treatment Effects	389
56.8 Insights into what makes treatment effects equal and $\mathcal{Y}_{1 0} = \mathcal{Y}_1$	392
56.9 G_{do+} bnet	393
56.10 $ACE = ATE$	394
56.11 Good, Bad Controls	395
56.12 PO Confounder Sensitivity Analysis	396
56.13 (SDO, ATE) space	397
56.14 Strata-Matching	400
56.14.1 Exact strata-matching	400
Estimates of Treatment Effects	400
Example, estimation of treatment effects	402
56.14.2 Approximate strata-matching	404
56.14.3 Unbiased strata-matching estimators	404
56.15 Propensities	406
56.16 Propensity based estimators of treatment effects	410
56.17 Positivity	411

56.18 Multi-time PO bnets (Panel Data)	412
57 Program evaluation and review technique (PERT)	416
57.1 Example	418
58 Random Forest and Bagging	422
58.1 Bagging (with fully-featured bags)	422
58.2 Bagging (with randomly-shortened bags)	424
59 Recurrent Neural Networks	425
59.1 Language Sequence Modeling	428
59.2 Other types of RNN	428
59.2.1 Long Short Term Memory (LSTM) unit (1997)	430
59.2.2 Gated Recurrence Unit (GRU) (2014)	432
60 Regression Discontinuity Design	434
60.1 PO analysis	434
60.2 Linear Regression	436
61 Reinforcement Learning (RL)	437
61.1 Exact RL bnet	440
61.2 Actor-Critic RL bnet	442
61.3 Q function learning RL bnet	444
62 Reliability Box Diagrams and Fault Tree Diagrams	446
62.1 Minimal Cut Sets	452
63 Restricted Boltzmann Machines	454
64 ROC curves	456
64.1 Terminology Table Adapted from Wikipedia Ref.[121]	459
65 Scoring the Nodes of a Learned Bnet	461
65.1 Probability Distributions and Special Functions	462
65.2 Single node with no parents	464
65.3 Multiple nodes with any number of parents	466
65.4 Bayesian Scores	468
65.5 Information Theoretic scores	468
66 Selection Bias Removal	470
66.1 Pre and Post Selection Nodes	471
66.2 Removing SB from passive query $P(y x)$	473
66.3 Removing SB from active query $P(y \mathcal{D}x)$	475

67 Sequential Backdoor Adjustment Formula	477
68 Shapley Explainability	481
68.0.1 Numerical examples of SHAP	484
69 Simpson's Paradox	487
69.1 Pearl Causality	489
69.2 Numerical Example	490
70 Structure and Parameter Learning for Bnets	491
70.1 Overview	491
70.2 Score based SL algorithms	493
70.3 Constraint based SL algorithms	494
70.4 Pseudo-code for some bnet learning algorithms	495
71 Support Vector Machines And Kernel Method	497
71.1 Learning Algorithm for SVM Classifier	498
71.2 Linear (dot-product) Kernel	499
71.3 Alternatives to Linear Kernel	503
71.4 Random Forest and Kernel Method	503
72 Synthetic Controls	504
72.1 PO analysis	506
73 Time Series Analysis: ARMA and VAR	508
73.1 White noise	508
73.2 Backshift operator	509
73.3 Metrics	509
73.4 Definition of $ARMA(p, q)$, $AR(p)$ and $MA(q)$	511
73.5 Solving $AR(p)$	513
73.6 Solving $MA(q)$	514
73.7 Solving $ARMA(p, q)$	515
73.8 Auto-correlation and partial auto-correlation	515
73.9 Generating function of auto-correlation	519
73.10 Impulse Response	520
73.11 $AR(p)$ and Yule-Walker equations	522
73.12 Forecasting	523
73.13 Model Learning	529
73.14 Differencing and $ARIMA(p, d, q)$	529
73.15 Parameter Learning	532
73.15.1 PL of $AR(p)$	533
73.15.2 PL of $MA(q)$	535
73.15.3 PL of $ARMA(p, q)$	537

73.16 $VAR(p)$	538
74 Transfer Learning	540
75 Transformer Networks	542
76 Transportability of Causal Knowledge	545
77 Turbo Codes	549
77.1 Decoding Algorithm	552
77.2 Message Passing Interpretation of Decoding Algorithm	554
78 Uplift Modelling	555
78.1 UP types	555
78.2 Some Relevant Technical Formulas from Chapter 56	556
78.3 UP Analysis	557
78.4 UP Decision Trees	559
78.4.1 Appendix, connection between Δ_c and $\Delta_{c j}$	564
79 Variational Bayesian Approximation	565
79.1 Free Energy $\mathcal{F}(\vec{x})$	567
80 XGBoost	570
80.1 Divergences	570
80.2 Minimizing Cost function for single tree	572
80.3 Leaf Splitting	575
80.4 Pruning	575
80.5 Feature Binning	577
80.6 Final estimate of target attribute	577
80.7 Bnet for XGBoost	578
81 Zero Information Transmission (Graphoid Axioms)	580
81.1 Consequences of Eq.(81.5)	581
Bibliography	583

Foreword

Welcome to Bayesuvius! a proto-book uploaded to github.

A different Bayesian network is discussed in each chapter. Each chapter title is the name of a Bnet. Chapter titles are in alphabetical order.

This is a volcano in its early stages. First version uploaded to a github repo called Bayesuvius on June 24, 2020. First version only covers 2 Bnets (Linear Regression and GAN). I will add more chapters periodically. Remember, this is a moon-lighting effort so I can't do it all at once.

For any questions about notation, please go to Notational Conventions section.
Requests and advice are welcomed.

Thanks for reading this

Robert R. Tucci

www.ar-tiste.xyz

ADDENDA

- **August 15, 2021:** At this point in time, the book has grown to 67 Chapters and 433 pages. Today, I am self-publishing it as an ebook at Amazon and similar outlets. It will still be free.

Navigating the ocean of Judea Pearl's Books

Many of the greatest ideas in the bnet field were invented by Judea Pearl and his collaborators. Thus, this book is heavily indebted to those great scientists. Those ideas have had no clearer and more generous expositor than Judea Pearl himself.

Pearl has written 4 books that I have used in writing Bayesuvius. His 1988 book Ref.[37] dates back to his pre-causal period. That book I used to learn about topics such as d-separation, belief propagation, Markov-blankets, and noisy-ORs. 3 other books that he wrote later, in his causal period, are:

1. In 2000 (1st ed.), and 2013 (2nd ed.), Pearl published what is so far his most technical and exhaustive book on the subject of causality, Ref.[39].
2. In 2016, he released together with Glymour and Jewell, a less advanced “primer” on causality, Ref.[42].
3. In 2018, he released together with Mackenzie his lovely “The Book of Why”, Ref.[43].

Those 3 books I used to learn about causality topics such as Do Calculus, backdoor and frontdoor adjustment formulae, linear deterministic bnets with exogenous noise, and counterfactuals.

A micro poem written by me to celebrate Judea Pearl and his work:

I, Robot

Let other robots `talk()`,
while I,
`talk()`, `do()` and `imagine()`.

CI-2-3 track

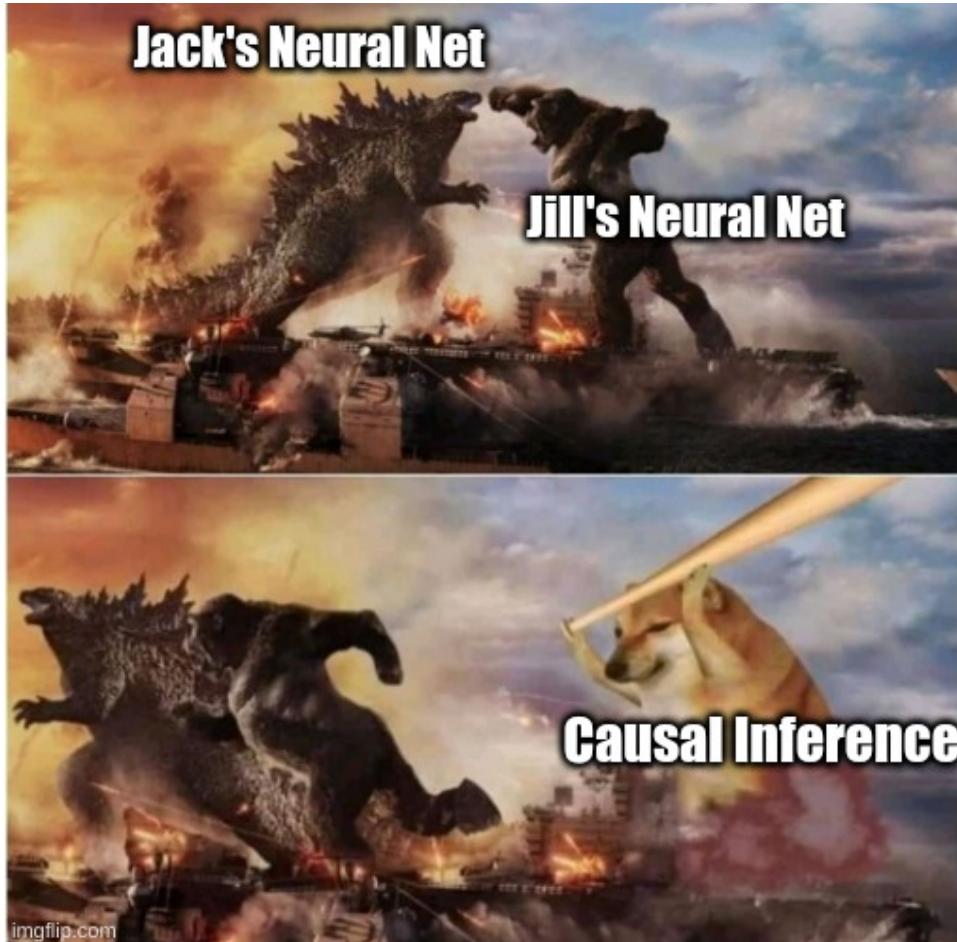


Figure 3: CI meme

As discussed in Chapter 10, Judea Pearl has proposed 3 rungs of Causal Inference (CI). This book covers all 3 rungs.

Confusingly, it has become common to use the term CI to refer to only the highest 2 rungs of the CI hierarchy; i.e, rung 2 (do operations) and rung 3 (imagining/counterfactual thinking). Also confusingly, rung 1 uses causal diagrams and is often referred to as “inference”, so it could reasonably have been defined as the

whole of CI, but Pearl has defined the CI hierarchy to include two more rungs. To patch over this linguistic confusion, I sometimes refer to rung 1 as “prediction”, or as “predictive inference” instead of calling it merely “inference”. Also, when I want to be precise, I use the term “CI-2-3” to refer to CI restricted to only rungs 2 and 3.

Here is a subset of chapters that I call the CI-2-3 track, that are devoted mostly to rungs 2 and 3.

1. Backdoor Adjustment Formula
2. Berkson’s Paradox
3. Counterfactual Reasoning
4. Decisions Based on Rungs 2 and 3: COMING SOON
5. Difference-in-Differences
6. Do Calculus
7. Do Calculus proofs
8. D-Separation
9. D-Separation and Do Calculus Rules for LDEN: COMING SOON
10. Frisch-Waugh-Lovell (FWL) theorem
11. Frontdoor Adjustment Formula
12. Goodness of Causal Fit
13. Granger Causality
14. Instrumental Inequality and beyond
15. Instrumental Variables
16. Mediation Analysis
17. Meta-learners for estimating ATE
18. Personalized Treatment Effects
19. Personalized Expected Utility
20. Potential Outcomes and Beyond
21. Regression Discontinuity Design
22. Sequential Backdoor Adjustment Formula

- 23. Selection Bias Removal
- 24. Simpson's Paradox
- 25. Synthetic Controls
- 26. Transportability of Causal Knowledge
- 27. Uplift Modelling

Notational Conventions and Preliminaries

0.1 Some abbreviations frequently used throughout this book

- bnet= Bnet= Bayesian Network
- CPT = Conditional Probabilities Table, same as TPM
- DAG = Directed Acyclic Graph
- i.i.d.= independent identically distributed.
- RCT= Randomized Controlled Trial, aka A/B testing.
- TPM= Transition Probability Matrix, same as CPT

0.2 $\mathcal{N}(!a)$

$\mathcal{N}(!a)$ will denote a normalization constant that does not depend on a . For example, $P(x) = \mathcal{N}(!x)e^{-x}$ where $\int_0^\infty dx P(x) = 1$.

0.3 One hot vector

A **one hot vector** is a vector with all entries equal to zero with the exception of a single entry which is one. A **one cold vector** is a vector with all entries equal to one with the exception of a single entry which is zero. For example, if $x^n = (x_0, x_1, \dots, x_{n-1})$ and $x_i = \delta(i, 0)$ then x^n is one hot.

Two types of sets that one frequently encounters are **categorical sets** (aka “nominal sets”, i.e., sets with “named” elements, with elements given a “nomme”) and **numerical sets** (aka “ordinal sets”, i.e., sets with “ordered” elements). For example, $\{1, 2, 5\}$ is a numerical set because its elements have a natural order, and $\{\text{cat, dog, bird}\}$ is a categorical set because its elements don’t have a natural order.

In Machine Learning (ML), one often encodes categorical sets as one-hot vectors. For example, suppose we have 4 binary registers (i.e., nodes) x_3, x_2, x_1, x_0 and the categorical set {cat, dog, canary}. Then a possible **one-hot encoding** of the set is cat=0001, dog=0010 and canary=0100. This differs from a **binary encoding** of the set such as cat=0000, dog=0001, canary=0011. Clearly, a binary encoding requires fewer registers than a one-hot encoding to encode the same set, and the one-hot encoding of a set with n elements requires n or more registers.

0.4 Special sets

Define $\mathbb{Z}, \mathbb{R}, \mathbb{C}$ to be the integers, real numbers and complex numbers, respectively.

For $a < b$, define \mathbb{Z}_I to be the integers in the interval I , where $I = [a, b], [a, b), (a, b], (a, b)$ (i.e, I can be closed or open on either side).

$$A_{>0} = \{k \in A : k > 0\} \text{ for } A = \mathbb{Z}, \mathbb{R}.$$

0.5 Kronecker delta function

For x, y in discrete set S ,

$$\delta(x, y) = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{if } x \neq y \end{cases} \quad (1)$$

0.6 Dirac delta function

For $x, y \in \mathbb{R}$,

$$\int_{-\infty}^{+\infty} dx \delta(x - y) f(x) = f(y) \quad (2)$$

0.7 Indicator function (aka Truth function)

$$\mathbb{1}(\mathcal{S}) = \begin{cases} 1 & \text{if } \mathcal{S} \text{ is true} \\ 0 & \text{if } \mathcal{S} \text{ is false} \end{cases} \quad (3)$$

For example, $\delta(x, y) = \mathbb{1}(x = y)$.

0.8 Majority function

The **majority function** is defined as follows.

$$\text{majority}(L) = \begin{array}{l} \text{most common element of list } L \\ (\text{ties resolved by chance}) \end{array} \quad (4)$$

Note that the majority function acts on lists, not sets. By definition, all elements of a set appear only once in the set. `majority`(L) is usually used when the elements of L are categorical (i.e., not real numbers). When they are real numbers, it makes more sense to use, instead of `majority`(L), a simple average of the elements of L .

0.9 Underlined letters indicate random variables

Random variables will be indicated by underlined letters and their values by non-underlined letters. Each node of a bnet will be labelled by a random variable. Thus, $\underline{x} = x$ means that node \underline{x} is in state x .

It is more conventional to use an upper case letter to indicate a random variable and a lower case letter for its state. Thus, $X = x$ means that random variable X is in state x . However, we have opted in this book to avoid that notation, because we often want to define certain lower case letters to be random variables or, conversely, define certain upper case letters to be non-random variables.

0.10 Probability distributions

$P_{\underline{x}}(x) = P(\underline{x} = x) = P(x)$ is the probability that random variable \underline{x} equals $x \in S_{\underline{x}}$. $S_{\underline{x}}$ is the set of states (i.e., values) that \underline{x} can assume and $n_{\underline{x}} = |S_{\underline{x}}|$ is the size (aka cardinality) of that set. Hence,

$$\sum_{x \in S_{\underline{x}}} P_{\underline{x}}(x) = 1 \quad (5)$$

$$P_{\underline{x}, \underline{y}}(x, y) = P(\underline{x} = x, \underline{y} = y) = P(x, y) \quad (6)$$

$$P_{\underline{x}|y}(x|y) = P(\underline{x} = x|\underline{y} = y) = P(x|y) = \frac{P(x, y)}{P(y)} \quad (7)$$

0.11 Discretization of continuous probability distributions

The TPM of a node of a bnet can be either a discrete or a continuous probability distribution. To go from continuous to discrete, one replaces integrals over states of a node by sums over new states, and Dirac delta functions by Kronecker delta functions. More precisely, consider a function $f : [a, b] \rightarrow \mathbb{R}$. Express $[a, b]$ as a union of small, disjoint (except for one point) closed sub-intervals (bins) of length Δx . Name one point in each bin to be the representative of that bin, and let $S_{\underline{x}}$ be the set of all the bin representatives. This is called discretization or binning. Then

$$\frac{1}{(b-a)} \int_{[a,b]} dx f(x) \rightarrow \frac{\Delta x}{(b-a)} \sum_{x \in S_{\underline{x}}} f(x) = \frac{1}{n_{\underline{x}}} \sum_{x \in S_{\underline{x}}} f(x) . \quad (8)$$

Both sides of last equation are 1 when $f(x) = 1$. Furthermore, if $y \in S_{\underline{x}}$, then

$$\int_{[a,b]} dx \delta(x-y)f(x) = f(y) \rightarrow \sum_{x \in S_{\underline{x}}} \delta(x,y)f(x) = f(y) . \quad (9)$$

0.12 Samples, i.i.d. variables

$$\vec{x} = (x[0], x[1], x[2] \dots, x[n sam(\vec{x}) - 1]) = x[:] \quad (10)$$

$n sam(\vec{x})$ is the number of samples of \vec{x} . $x[\sigma] \in S_{\underline{x}}$ are i.i.d. (independent identically distributed) samples with

$$x[\sigma] \sim P_{\underline{x}} \text{ (i.e. } P_{x[\sigma]} = P_{\underline{x}}) \quad (11)$$

$$P(\underline{x} = x) = \frac{1}{n sam(\vec{x})} \sum_{\sigma} \mathbb{1}(x[\sigma] = x) \quad (12)$$

Hence, for any $f : S_{\underline{x}} \rightarrow \mathbb{R}$,

$$\sum_x P(\underline{x} = x) f(x) = \frac{1}{n sam(\vec{x})} \sum_{\sigma} f(x[\sigma]) \quad (13)$$

If we use two sampled variables, say \vec{x} and \vec{y} , in a given bnet, their number of samples $n sam(\vec{x})$ and $n sam(\vec{y})$ need not be equal.

$$P(\vec{x}) = \prod_{\sigma} P(x[\sigma]) \quad (14)$$

$$\sum_{\vec{x}} = \prod_{\sigma} \sum_{x[\sigma]} \quad (15)$$

$$\partial_{\vec{x}} = [\partial_{x[0]}, \partial_{x[1]}, \partial_{x[2]}, \dots, \partial_{x[n sam(\vec{x}) - 1]}] \quad (16)$$

$$P(\vec{x}) \approx [\prod_x P(x)^{P(x)}]^{n sam(\vec{x})} \quad (17)$$

$$= e^{n sam(\vec{x}) \sum_x P(x) \ln P(x)} \quad (18)$$

$$= e^{-n sam(\vec{x}) H(P_{\underline{x}})} \quad (19)$$

0.13 Expected Value and Variance

Given a random variable \underline{x} with states $S_{\underline{x}}$ and a function $f : S_{\underline{x}} \rightarrow \mathbb{R}$, define

$$E_{\underline{x}}[f(\underline{x})] = E_{x \sim P(x)}[f(x)] = \sum_x P(x)f(x) \quad (20)$$

$$Var_{\underline{x}}[f(\underline{x})] = E_{\underline{x}}[(f(\underline{x}) - E_{\underline{x}}[f(\underline{x})])^2] \quad (21)$$

$$= E_{\underline{x}}[f(\underline{x})^2] - (E_{\underline{x}}[f(\underline{x})])^2 \quad (22)$$

$$E[\underline{x}] = E_{\underline{x}}[\underline{x}] \quad (23)$$

$$Var[\underline{x}] = Var_{\underline{x}}[\underline{x}] \quad (24)$$

0.14 Conditional Expected Value

Given a random variable \underline{x} with states $S_{\underline{x}}$, a random variable \underline{y} with states $S_{\underline{y}}$, and a function $f : S_{\underline{x}} \times S_{\underline{y}} \rightarrow \mathbb{R}$, define

$$E_{\underline{x}|\underline{y}}[f(\underline{x}, \underline{y})] = \sum_x P(x|\underline{y})f(x, \underline{y}), \quad (25)$$

$$E_{\underline{x}|y=y}[f(\underline{x}, y)] = E_{\underline{x}|y}[f(\underline{x}, y)] = \sum_x P(x|y)f(x, y). \quad (26)$$

Note that

$$E_{\underline{y}}[E_{\underline{x}|\underline{y}}[f(\underline{x}, \underline{y})]] = \sum_{x,y} P(x|y)P(y)f(x, y) \quad (27)$$

$$= \sum_{x,y} P(x, y)f(x, y) \quad (28)$$

$$= E_{\underline{x}, \underline{y}}[f(\underline{x}, \underline{y})]. \quad (29)$$

0.15 Notation for covariances

Consider two random variables $\underline{x}, \underline{y}$.

- Mean value of \underline{x}

$$\langle \underline{x} \rangle = E_{\underline{x}}[\underline{x}] \quad (30)$$

- Signed distance of \underline{x} to its mean value

$$\Delta \underline{x} = \underline{x} - \langle \underline{x} \rangle \quad (31)$$

- Covariance of $(\underline{x}, \underline{y})$

$$Cov(\underline{x}, \underline{y}) = \langle \underline{x}, \underline{y} \rangle = \langle \Delta \underline{x} \Delta \underline{y} \rangle = \langle \underline{x} \underline{y} \rangle - \langle \underline{x} \rangle \langle \underline{y} \rangle \quad (32)$$

$\langle \underline{x}, \underline{y} \rangle$ is symmetric (i.e., $\langle \underline{x}, \underline{y} \rangle = \langle \underline{y}, \underline{x} \rangle$) and bilinear (i.e., $\langle \sum_i \alpha_i \underline{x}_i, \underline{y} \rangle = \sum_i \alpha_i \langle \underline{x}_i, \underline{y} \rangle$, where $\alpha_i \in \mathbb{R}$ are non-random scalars and $\underline{x}_i, \underline{y} \in \mathbb{R}$ are real-valued random variables.)

- Variance of \underline{x}

$$Var(\underline{x}) = \langle \underline{x}, \underline{x} \rangle \quad (33)$$

- Standard deviation or \underline{x}

$$\sigma_{\underline{x}} = \sqrt{\langle \underline{x}, \underline{x} \rangle} \quad (34)$$

- Correlation Coefficient of $(\underline{x}, \underline{y})$

$$\rho_{\underline{x}, \underline{y}} = \frac{\langle \underline{x}, \underline{y} \rangle}{\sqrt{\langle \underline{x}, \underline{x} \rangle \langle \underline{y}, \underline{y} \rangle}} \quad (35)$$

0.16 Conditional Covariance

Let $\underline{x}, \underline{y}, \underline{a}$ be random variables. The covariance $Cov(\underline{x}, \underline{y}|\underline{a})$ of \underline{x} and \underline{y} given \underline{a} , is defined the same way as $Cov(\underline{x}, \underline{y})$, except that all expected values are conditioned on \underline{a} .

$$Cov(\underline{x}, \underline{y}|\underline{a}) = \langle \underline{x}, \underline{y} \rangle_{|\underline{a}} = \left\langle (\underline{x} - \langle \underline{x} \rangle_{|\underline{a}})(\underline{y} - \langle \underline{y} \rangle_{|\underline{a}}) \right\rangle_{|\underline{a}} \quad (36)$$

where

$$\langle \underline{x} \rangle_{|\underline{a}} = E_{\underline{x}|\underline{a}}[\underline{x}] . \quad (37)$$

0.17 Law of Total Variance

Claim 1 Suppose $P : S_{\underline{x}} \times S_{\underline{y}} \rightarrow [0, 1]$ is a probability distribution. Suppose $f : S_{\underline{x}} \times S_{\underline{y}} \rightarrow \mathbb{R}$ and $f = f(x, y)$. Then

$$Var_{\underline{x}, \underline{y}}(f) = E_{\underline{y}}[Var_{\underline{x}|\underline{y}}(f)] + Var_{\underline{y}}(E_{\underline{x}|\underline{y}}[f]) . \quad (38)$$

In particular,

$$Var_{\underline{x}}(x) = E_{\underline{y}}[Var_{\underline{x}|\underline{y}}(x)] + Var_{\underline{y}}(E_{\underline{x}|\underline{y}}[x]) . \quad (39)$$

proof:

Let

$$A = \sum_y P(y) \left(\sum_x P(x|y)f \right)^2. \quad (40)$$

Then

$$Var_{\underline{x},\underline{y}}(f) = \sum_{x,y} P(x,y)f^2 - \left(\sum_{x,y} P(x,y)f \right)^2 \quad (41)$$

$$= \begin{cases} \sum_{x,y} P(x,y)f^2 - A \\ + \left(A - \left(\sum_{x,y} P(x,y)f \right)^2 \right) \end{cases} \quad (42)$$

$$E_{\underline{y}}[Var_{\underline{x}|\underline{y}}(f)] = \sum_y P(y) \left(\sum_x P(x|y)f^2 - \left(\sum_x P(x|y)f \right)^2 \right) \quad (43)$$

$$= \sum_{x,y} P(x,y)f^2 - A \quad (44)$$

$$Var_{\underline{y}}(E_{\underline{x}|\underline{y}}[f]) = \sum_y P(y) \left(\sum_x P(x|y)f \right)^2 - \left(\sum_y P(y) \sum_x P(x|y)f \right)^2 \quad (45)$$

$$= A - \left(\sum_{x,y} P(x,y)f \right)^2 \quad (46)$$

QED

0.18 Normal Distribution

For $x, \mu, \sigma \in \mathbb{R}$, $\sigma > 0$, we define the Normal Distribution (see Fig.4) by

$$\mathcal{N}(x; \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}. \quad (47)$$

For a **standard deviation** σ , the **precision** τ is defined as $\tau = \frac{1}{\sigma^2}$.

Claim 2 If

$$\underline{x}_1 \sim \mathcal{N}(\mu_1, \sigma_1^2) \quad (48)$$

and

$$\underline{x}_2 \sim \mathcal{N}(\mu_2, \sigma_2^2) \quad (49)$$

then

$$\underline{x} = \underline{x}_1 + \underline{x}_2 \sim \mathcal{N}(\mu_1 + \mu_2, \sigma_1^2 + \sigma_2^2). \quad (50)$$

proof:

$$P(\underline{x} = x) = \mathcal{N}(!x) \int_{-\infty}^{+\infty} dx_2 P(\underline{x}_1 + \underline{x}_2 = x | \underline{x}_2 = x_2) P(x_2) \quad (51)$$

$$= \mathcal{N}(!x) \int_{-\infty}^{+\infty} dx_2 \mathcal{N}(x - x_2; \mu_1, \sigma_1^2) \mathcal{N}(x_2; \mu_2, \sigma_2^2) \quad (52)$$

$$= \mathcal{N}(x; \mu_1 + \mu_2; \sigma_1^2 + \sigma_2^2) \quad (53)$$

QED

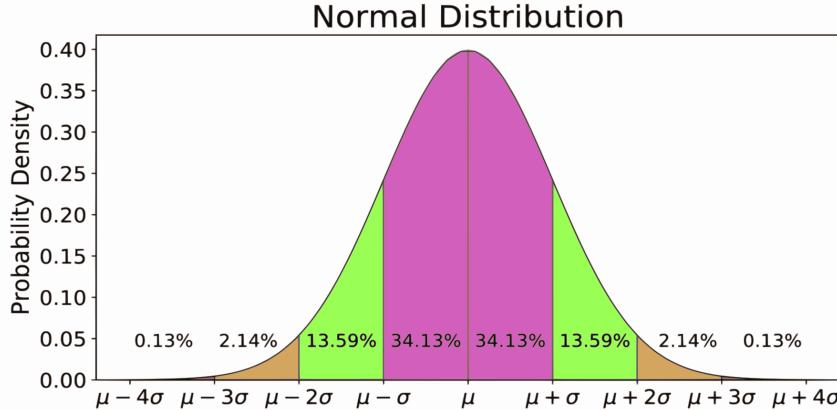


Figure 4: Normal Distribution $\mathcal{N}(x; \mu, \sigma^2)$.

The **Standard Normal Distribution** $P_{SND}(x)$ and its cumulative distribution $\Phi(x)$ are defined by

$$P_{SND}(x) = \mathcal{N}(x; \mu = 0, \sigma = 1) \quad (54)$$

$$\Phi(x) = \int_{-\infty}^x dx' P_{SND}(x') \quad (55)$$

The **error function** $\text{erf} : \mathbb{R} \rightarrow [-1, 1]$ is defined by

$$\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x du e^{-\frac{u^2}{2}} \quad (56)$$

Note that

$$\Phi(x) = \frac{1}{2} + \frac{1}{2}\text{erf}(x) \quad (57)$$

Eq.(57) is interpreted geometrically in Fig.5.

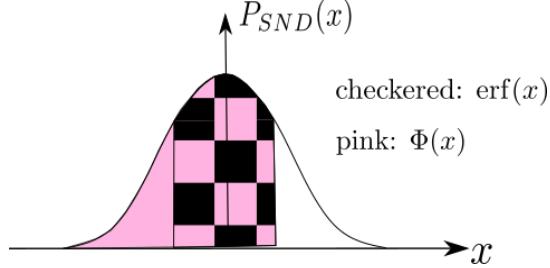


Figure 5: Plot of Standard Normal Distribution $P_{SND}(x)$. Values of $\text{erf}(x)$ and $\Phi(x)$ equal indicated areas.

0.19 Uniform Distribution

For $a < b$, $x \in [a, b]$

$$\mathcal{U}(x; a, b) = \frac{1}{b - a} \quad (58)$$

0.20 Softmax function (aka Boltzmann Distribution)

The Softmax function is defined by

$$P(x_i|x.) = \frac{e^{x_i}}{\sum_i e^{x_i}} = \text{softmax}(x.)_i \quad (59)$$

The Boltzmann distribution is defined as

$$P(E_a = E_a) = \frac{\exp(-\frac{E_a}{kT})}{\sum_a \exp(-\frac{E_a}{kT})} = P(\frac{-E_a}{kT}|E.) \quad (60)$$

for a system with energies E_a and temperature T , where k is Boltzmann's constant.

The function softmax() is called softmax because if we approximate the exponentials, both in the numerator and denominator of Eq.(59), by the largest one of them or zero, we get

$$\text{softmax}(x.)_i \approx \mathbb{1}(i = \operatorname{argmax}_k x_k) . \quad (61)$$

Thus, $\text{softmax}(x.)$ returns a continuous function that approximates a one-hot vector that is 1 at the i th component, where $i = \underset{k}{\text{argmax}}(x_k)$, and zero at the other components.

Note that

$$\frac{\partial \ln P(x_i|x)}{\partial x_a} = \frac{\partial}{\partial x_a} \ln \left[\frac{e^{x_i}}{\sum_i e^{x_i}} \right] = \delta(a, i) - P(x_a|x) \quad (62)$$

For 2 variables x_0, x_1 ,

$$P(x_0|x.) = \frac{e^{x_0}}{e^{x_0} + e^{x_1}} \quad (63)$$

$$= \text{smoid}(x_0 - x_1), \quad (64)$$

$$P(x_1|x.) = \text{smoid}(x_1 - x_0). \quad (65)$$

0.21 Sigmoid and log-odds functions

The **sigmoid (aka exp-it, logistic) function** $\text{smoid}: \mathbb{R} \rightarrow [0, 1]$ is defined by

$$\text{smoid}(x) = \frac{1}{1 + e^{-x}} \quad (66)$$

$\text{smoid}()$ is monotonically increasing with $\text{smoid}(-\infty) = 0$, $\text{smoid}(0) = 1/2$ and $\text{smoid}(+\infty) = 1$.

$$\text{smoid}(x) + \text{smoid}(-x) = \frac{1}{1 + e^{-x}} + \frac{1}{1 + e^x} \quad (67)$$

$$= \frac{2 + e^x + e^{-x}}{2 + e^x + e^{-x}} \quad (68)$$

$$= 1 \quad (69)$$

The **log-odds (aka log-it) function** $\text{lodds}:[0, 1] \rightarrow \mathbb{R}$ is defined by

$$\text{lodds}(p) = \ln \frac{p}{1-p} \quad (70)$$

$\text{lodds}()$ is the inverse of $\text{smoid}()$ and vice-versa. For $p \in [0, 1]$ and $x \in \mathbb{R}$,

$$\text{lodds}[\text{smoid}(x)] = x \quad (71)$$

$$\text{smoid}[\text{lodds}(p)] = p \quad (72)$$

Claim 3

$$\text{smoid}'(x) = \text{smoid}(x)[1 - \text{smoid}(x)] \quad (73)$$

$$\text{smoid}''(x) = \text{smoid}'(x)[1 - 2\text{smoid}(x)] \quad (74)$$

proof:

In this proof, we will abbreviate $\text{smoid}(x)$ by $s(x)$.

$$1 - s(x) = 1 - \frac{1}{1 + e^{-x}} = \frac{e^{-x}}{1 + e^{-x}} \quad (75)$$

$$s'(x) = \frac{e^{-x}}{(1 + e^{-x})^2} = s(x)[1 - s(x)] \quad (76)$$

$$s''(x) = s'(x)[1 - s(x)] + s(x)(-1)s'(x) \quad (77)$$

$$= s'(x)[1 - 2s(x)] \quad (78)$$

$$= s(x)[1 - s(x)][1 - 2s(x)] \quad (79)$$

QED

0.22 Estimand, Estimator, Bias

We say $\hat{\theta}$ is an **estimator** (aka **estimate**) of a quantity θ if $E[\hat{\theta}] = \theta + b$ for some **bias** b . We say this estimator is an **unbiased estimator** if $b = 0$. θ is sometimes called **the estimand** of $\hat{\theta}$.

0.23 Maximum Likelihood Estimate, Likelihood Ratio Test

Given a bnet, let $P(x|\theta)$ be its full joint probability distribution, where x denotes the joint state of all the nodes and θ denotes all the parameters. $P(x|\theta)$ is often called the **likelihood function of θ** and is denoted by

$$L(\theta) = P(x|\theta) \quad (80)$$

It's called a likelihood of θ because, even though it's a probability, it isn't the probability of θ , but rather of x .

The value of θ that we obtain by maximizing $L(\theta)$ over θ is called the **maximum likelihood estimate (MLE) of θ** . Let us denote it by $\hat{\theta}$. Note that¹

$$\sup_{\theta \in S} L(\theta) = L(\hat{\theta}) \quad (81)$$

Let S_0, S_1 be disjoint sets such that $S = S_0 \cup S_1$. We'll say the **null hypothesis** H_0 holds if $\theta \in S_0$, and the **alternative hypothesis** H_1 holds if $\theta \in S_1$. The **likelihood ratio (LR) test statistic** is defined by

$$R = -2 \ln \left(\frac{\sup_{\theta \in S_0} L(\theta)}{\sup_{\theta \in S} L(\theta)} \right) \quad (82)$$

$R \geq 0$ and $R = 0$ if $S_0 = S$. For some small $c > 0$, if $R < c$, then we reject the alternative hypothesis, and if $R > c$, we accept it.

If $S_0 = \{\theta_0\}$, then

$$R = -2[\ln L(\theta_0) - \ln L(\hat{\theta})] \quad (83)$$

0.24 Mean Square Error (MSE)

Suppose we are given $nsam$ samples $y^\sigma \in \mathbb{R}$ labeled by an index σ , and an **estimator** function $\hat{y}^\sigma(a) \in \mathbb{R}$ that depends on a parameter $a \in \mathbb{R}$. Define the **Mean Square Error (MSE)** by

$$MSE(a) = \frac{1}{nsam} \sum_{\sigma} (y^\sigma - \hat{y}^\sigma(a))^2. \quad (84)$$

For example, in Linear Regression (LR), we have $\hat{y}^\sigma = a_0 + a_1 x^\sigma$ where $a = (a_0, a_1)$ is a deterministic parameter. If the samples y^σ are i.i.d, then we can also write

$$MSE(a) = E_{|a}[(\underline{y} - \hat{y}(a))^2]. \quad (85)$$

and for LR, $\hat{y}(a) = a_0 + a_1 \underline{x}$.

Define the **residual** $\Delta\underline{y}$ by:

$$\Delta\underline{y}(a) = \underline{y} - \hat{y}(a) \quad (\text{Hence } \underline{y} = \hat{y} + \Delta\underline{y}) \quad (86)$$

In the rest of this section, we will discuss the case that $\hat{y}^\sigma(a)$ is independent of x^σ . I call this the **deterministic MSE (D-MSE)** model. Note that this is different from the LR case where $\hat{y}^\sigma(a)$ does depend on x^σ . In LR, we are trying to fit a line to a cigar-shaped 2-D scatter plot. Here, we are just trying to estimate the mean value (center of mass) of a scatter plot.

¹“sup” stands for supremum. It's a generalization of the function $\max()$ to arbitrary sets that might not be discrete or finite. If S is a finite set, then $\sup_{\theta \in S} f(\theta) = \max_{\theta \in S} f(\theta)$ for any function $f : S \rightarrow \mathbb{R}$. Likewise, “inf” stands for infimum, and it generalizes the $\min()$ function.

Claim 4 *MSE is minimized over all functions \hat{y} if*

$$\hat{y} = E_{|a}[\underline{y}] \quad (87)$$

proof:

$$MSE = E_{|a}[\underline{y}^2] - 2\hat{y}E_{|a}[\underline{y}] + \hat{y}^2 \quad (88)$$

$$0 = \frac{d}{d\hat{y}}MSE = 2(-E_{|a}[\underline{y}] + \hat{y}) \quad (89)$$

Hence,

$$\hat{y} = E_{|a}[\underline{y}] \quad (90)$$

QED

Sometimes, we will use the notation

$$\hat{y}_{MSE} = E_{|a=a_{MSE}}[\underline{y}] . \quad (91)$$

Claim 5 *Suppose $f(a)$ is a function of a . If $\hat{y} = E_{|a}[\underline{y}]$, then*

$$E_{|a}[\Delta\underline{y}] = E[\Delta\underline{y}] = 0 \quad (92)$$

$$E_{|a}[\Delta\underline{y}f(\underline{a})] = E[\Delta\underline{y}f(\underline{a})] = 0 \quad (93)$$

proof:

$$E_{|a}[\Delta\underline{y}] = E_{|a}[y - E_{|a}[\underline{y}]] = E_{|a}[y] - E_{|a}[\underline{y}] = 0 \quad (94)$$

$$E[\Delta\underline{y}] = E_a[E_{|a}[\Delta\underline{y}]] = 0 \quad (95)$$

$$E_a[\Delta\underline{y}f(\underline{a})] = f(\underline{a}) \underbrace{E_{|a}[\Delta\underline{y}]}_{=0} \quad (96)$$

$$E[\Delta\underline{y}f(\underline{a})] = E_a[E_{|a}[\Delta\underline{y}f(\underline{a})]] = 0 \quad (97)$$

QED

Claim 6 If $\hat{y} = E_{|a}[\underline{y}]$, then

$$\langle \Delta\underline{y}, \hat{y} \rangle_{|a} = 0 \quad (98)$$

$$Var_{|a}[\underline{y}] = Var_{|a}[\hat{y}] + Var_{|a}[\Delta\underline{y}] \quad (99)$$

The same results hold without the conditioning on a .

proof:

$$\langle \Delta\underline{y}, \hat{y} \rangle_{|a} = \underbrace{E_{|a}[\Delta\underline{y}] \underbrace{\hat{y}}_{f(a)}}_{=0} - \underbrace{E_{|a}[\Delta\underline{y}]}_{=0} E_{|a}[\hat{y}] \quad (100)$$

$$Var_{|a}[\underline{y}] = \langle \hat{y} + \Delta\underline{y}, \hat{y} + \Delta\underline{y} \rangle_{|a} \quad (101)$$

$$= \langle \hat{y}, \hat{y} \rangle_{|a} + \langle \Delta\underline{y}, \Delta\underline{y} \rangle_{|a} \quad (\text{by Eq.(98)}) \quad (102)$$

$$= Var_{|a}[\hat{y}] + Var_{|a}[\Delta\underline{y}] \quad (103)$$

The same proof holds if we remove all the $|a$ subscripts.

QED

Fig.6 illustrates how $\underline{y} = \hat{y} + \Delta\underline{y}$ and the variances of these quantities add.

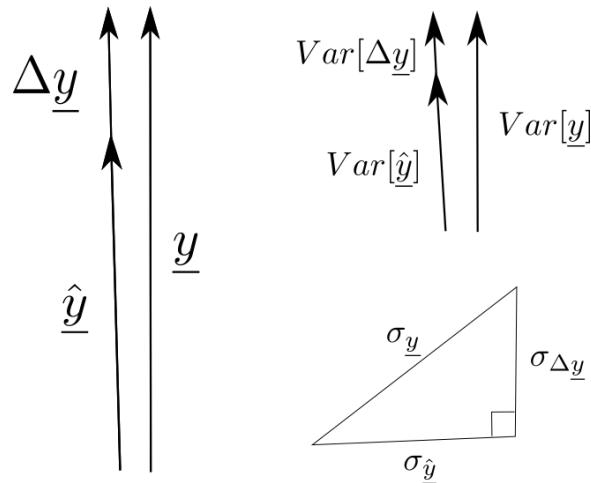


Figure 6: $\underline{y} = \hat{y} + \Delta\underline{y}$ and the variances (not standard deviations) of these quantities add.

0.25 Bayes Rule, Bayesian Updating And Conjugate Priors

Bayes Rule says for $x = (x_1, x_2)$:

$$P(\theta|x)P(x) = P(x|\theta)P(\theta)$$

$$\theta \begin{array}{c} \nearrow x_1 \\ \downarrow \\ \searrow x_2 \end{array} \quad P(x) \quad = \quad \theta \begin{array}{c} \nearrow x_1 \\ \downarrow \\ \searrow x_2 \end{array} \quad P(\theta) \quad (104)$$

Note how Bayes rule allows us to reverse the direction of the arrows impinging on θ . We see from Bayes Rule that even though the directions of the arrows in a bnet can have causal motivation, a bnet with arrows reversed from their causally motivated directions can still be very useful as a calculation tool.

Another way of stating Bayes Rule is

$$\underbrace{P(\theta|x)}_{\text{posterior}} = \mathcal{N}(!\theta) \underbrace{P(x|\theta)}_{\text{likelihood}} \underbrace{P(\theta)}_{\text{prior}} . \quad (105)$$

If, for a given likelihood, the prior and posterior distributions belong to the same family (for instance, they are both Beta distributions), then we say that the prior is the **conjugate prior** of that likelihood.

For example, $\text{Beta} \sim \text{Bernoulli}^*\text{Beta}$. Hence, the Beta distribution² is the conjugate prior of the Bernoulli distribution³. More explicitly, if

$$p_1 \sim \text{Beta}(p_1; \alpha, \beta) \quad (106)$$

and

$$x|p_1 \sim \text{Bernoulli}(x; p_1) , \quad (107)$$

where $p_1 = P(x = 1)$, then

$$p_1|x \sim \text{Beta}(p_1; \alpha', \beta') \quad (108)$$

where

$$\alpha' = \alpha + x \quad (109)$$

$$\beta' = \beta + (1 - x) \quad (110)$$

Ref.[81] has a table of conjugate priors.

²See Ref.[72] for a discussion of the Beta distribution.

³See Ref.[71] for a discussion of the Bernoulli distribution

Conjugate priors facilitate Bayesian updating of the prior to posterior in a feedback loop(see Fig.7).

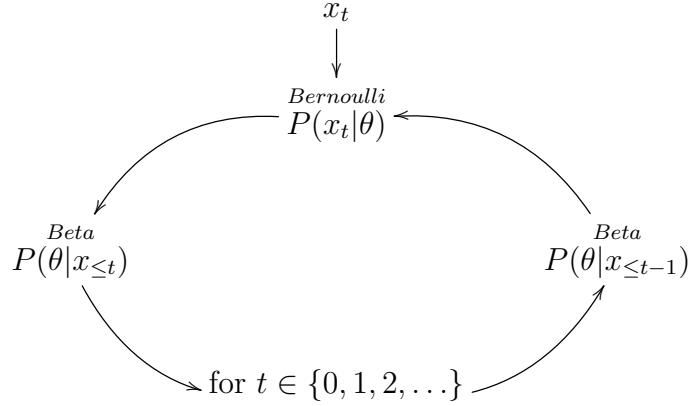


Figure 7: Bayesian updating facilitated by conjugate prior. In this figure, $x_{\leq t} = (x_0, x_1, \dots, x_{t-1}, x_t)$.

0.26 Linear regression, Ordinary Least Squares (OLS)

Wikipedia articles

1. Linear Regression (LR)

- linear regression, Ref.[106]
- simple linear regression, Ref.[124]
- errors in variable, Ref.[85]

2. Least squares (LS)

- least squares, Ref.[104]
- ordinary least squares (OLS), Ref.[118]

Some nomenclature: In LR, the data consists of **independent x-variables** $x_1^\sigma, x_2^\sigma, \dots, x_n^\sigma$ and a **dependent y-variable** y^σ . We find a linear fit $\hat{y}^\sigma = \beta_0 + \sum_{i=1}^n \beta_i x_i^\sigma$ to the data. \hat{y}^σ is called the **estimate** of y^σ . The coefficients β_0, β_i are called **regression coefficients**. $y^\sigma - \hat{y}^\sigma = \epsilon^\sigma$ are called the **residuals**. $\mathcal{E} = \sum_\sigma (\epsilon^\sigma)^2$ is called the **error or cost**. We choose the regression coefficients so as to minimize the error.

Below, we consider two types of LR:

1. LR in which the independent x-variables are non-random.

2. LR in which the independent x-variables are random and i.i.d.

The term OLS is often used to refer to LR of type 1.

For LR of type 2, there is randomness in y coming from the randomness in x and in the residuals. For LR of type 1, there is randomness in y too, but it comes from the residuals only.

Once one assumes that certain variables are random, a “model” (i.e., a bnet, with probabilities expressed as TPMs) must be specified.

0.26.1 LR, assuming x_σ are non-random

Let

- $\sigma \in \{0, 1, 2, \dots, nsam - 1\}$: sample index
- $i_0 \in \{0, 1, 2, \dots, n\}$: index that can assume values 0 to n
- $i \in \{1, 2, \dots, n\}$: index that can assume values 1 to n . i is never equal to 0.
- $y_\sigma \in \mathbb{R}$: dependent y-variables
- $x_{\sigma i} \in \mathbb{R}$: independent x-variables
- $\epsilon_\sigma \in \mathbb{R}$: residuals
- $\beta_0, \beta_i \in \mathbb{R}$: regression coefficients

$$y_\sigma = \beta_0 + \sum_{i=1}^n x_{\sigma i} \beta_i + \epsilon_\sigma \quad (111)$$

If we define

$$x_{\sigma 0} = 1 \quad (112)$$

for all σ , then

$$y_\sigma = \sum_{i_0=0}^n x_{\sigma i_0} \beta_{i_0} + \epsilon_\sigma . \quad (113)$$

If y and ϵ are $nsam$ dimensional column vectors and β is an $n+1$ dimensional column vector, and X is an $nsam \times (n+1)$ matrix, then we can write the previous equation in matrix form as:

$$y = X\beta + \epsilon . \quad (114)$$

Derivation of LR From Minimization of Error

Let $W = [W_{\sigma, \sigma'}]$ be a symmetric matrix with non-negative diagonal elements $W_{\sigma, \sigma} \geq 0$ for all σ . W is called the **weight matrix**. The following claim describes the method of **Weighted LR** when $W \neq 1$ and of simple LR when $W = 1$.

Claim 7 Assume the Einstein summation convention; i.e., implicit sum over repeated indices. The error function \mathcal{E} given by

$$\mathcal{E} = \underbrace{(y_\sigma - X_{\sigma,j_0}\beta_{j_0})}_{\text{residual } \epsilon_\sigma} W_{\sigma,\sigma'} \underbrace{(y_{\sigma'} - X_{\sigma',k_0}\beta_{k_0})}_{\epsilon_{\sigma'}}, \quad (115)$$

is minimized over β_{k_0} for all $k_0 \in \{0, 1, \dots, n\}$, if β_{k_0} is given by:

$$\hat{\beta} = (X^T W X)^{-1} X^T W y. \quad (116)$$

When $W = 1$,

$$\hat{\beta} = (X^T X)^{-1} X^T y. \quad (117)$$

proof:

At the minimum of \mathcal{E} , the variation $\delta\mathcal{E}$ must vanish:

$$0 = \delta\mathcal{E} = -2X_{\sigma j_0}(\delta\beta_{j_0})W_{\sigma,\sigma'}(y_{\sigma'} - X_{\sigma' k_0}\beta_{k_0}). \quad (118)$$

Thus,

$$X^T W y - X^T W X \beta = 0 \quad (119)$$

which implies Eq.(116).

QED

Geometry of LR with non-random x_σ .

Recall that

$$y = X\beta + \epsilon. \quad (120)$$

Define the **projection matrices**

$$\wedge = X(X^T X)^{-1} X^T, \quad \vee = 1 - \wedge \quad (121)$$

A square matrix M is symmetric if $M^T = M$ and is idempotent if $M^2 = M$. \wedge is symmetric and idempotent and so is \vee . Note that \wedge and \vee also satisfy:

$$\vee\wedge = \wedge\vee = 0 \quad (122)$$

and

$$\wedge X = X, \quad \vee X = 0. \quad (123)$$

One has

$$\beta = (X^T X)^{-1} X^T (y - \epsilon). \quad (124)$$

Define

$$\hat{\beta} = \underbrace{(X^T X)^{-1} X^T}_{B} y , \quad (125a)$$

$$\hat{y} = X\hat{\beta} = \wedge y , \quad (125b)$$

and

$$\hat{\epsilon} = y - X\hat{\beta} = y - \hat{y} = (1 - \wedge)y = \vee y . \quad (125c)$$

\wedge is sometimes called the **hat matrix**, because it gives y a hat.

Given any function $f = f(y, X, \epsilon)$ and a scalar factor $\xi \in \mathbb{R}$, suppose $f(\xi y, \xi X, \xi \epsilon) = \xi^{\mathcal{O}} f(y, X, \epsilon)$. Then we will say that $f(\cdot)$ is of **order \mathcal{O} under scaling**. Note that $\{\hat{y}, \hat{\epsilon}\}$ are all of order 1 under scaling, $\{\beta, \hat{\beta}, \wedge, \vee\}$ are all of order 0 under scaling, and B is of order -1 under scaling. Thus, each estimator (i.e., symbol with a hat) scales the same way as its estimand (i.e., same symbol without a hat). Furthermore, β , its estimator $\hat{\beta}$, and the projection matrices \wedge, \vee are invariant ($\mathcal{O} = 0$) under scaling.

Note that y can be expressed as a sum of 2 orthogonal estimators:

$$y = \underbrace{\hat{y}}_{\wedge y} + \underbrace{\hat{\epsilon}}_{\vee y} . \quad (126)$$

Fig.8 shows triangles representing $y = X\beta + \epsilon$ and $y = \hat{y} + \hat{\epsilon}$.

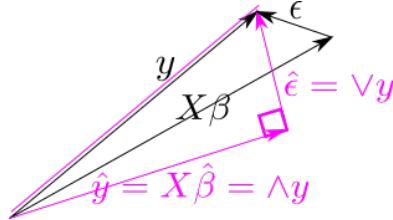


Figure 8: Triangles representing $y = X\beta + \epsilon$ and $y = \hat{y} + \hat{\epsilon}$.

LR Goodness of Fit, R^2

Assume the components of ϵ are random with zero mean:

$$E[\epsilon] = \langle \epsilon \rangle = 0 \quad (127)$$

Assume X and β are not random. This makes $\underline{y} = X\beta + \underline{\epsilon}$ and $\underline{\beta} = (X^T X)^{-1} X^T \underline{y}$ random. One finds that

$$\langle \underline{y} \rangle = X\beta \quad (128a)$$

$$\langle \hat{y} \rangle = \wedge \underbrace{\langle y \rangle}_{X\beta} = \langle y \rangle \quad (128b)$$

$$\langle \hat{\epsilon} \rangle = \vee \underbrace{\langle y \rangle}_{X\beta} = 0 \quad (128c)$$

$$\langle \hat{\beta} \rangle = \beta \quad (128d)$$

So far, we have assumed a zero mean value for ϵ . Next, assume “**homoscedasticity**” (**homo-spread**)⁴, which means that

$$\langle \epsilon, \epsilon^T \rangle = \xi^2 I_{nsam} \quad (128e)$$

where $\xi \geq 0$, and I_{nsam} is the $nsam \times nsam$ identity matrix. It follows that

$$\langle y, y^T \rangle = \langle \epsilon, \epsilon^T \rangle = \xi^2 I_{nsam}, \quad (129a)$$

$$\langle \hat{\epsilon}, \hat{\epsilon}^T \rangle = \vee \langle y, y^T \rangle \vee^T = \xi^2 \vee, \quad (129b)$$

$$\langle \hat{y}, \hat{y}^T \rangle = \wedge \langle y, y^T \rangle \wedge^T = \xi^2 \wedge \quad (129c)$$

and

$$\langle \hat{\beta}, \hat{\beta}^T \rangle = B \langle y, y^T \rangle B^T = \xi^2 (X^T X)^{-1}. \quad (129d)$$

For any random column vector \underline{a} , let

$$\| \underline{a} \|^2 = \underline{a}^T \underline{a} = \text{tr}(\underline{a} \underline{a}^T) \quad (130)$$

and

$$\langle \| \underline{a} - \langle \underline{a} \rangle \|^2 \rangle = \langle \underline{a}^T \underline{a} \rangle - \langle \underline{a}^T \rangle \langle \underline{a} \rangle = \text{tr} \langle \underline{a}, \underline{a}^T \rangle. \quad (131)$$

Define the following sums of squares (SS):

$$SS_y = \langle \| y - \langle y \rangle \|^2 \rangle = \langle y^T y \rangle - \langle y^T \rangle \langle y \rangle = \text{tr} \langle y, y^T \rangle \quad (132a)$$

$$SS_{\hat{y}} = \langle \| \hat{y} - \langle \hat{y} \rangle \|^2 \rangle = \langle \hat{y}^T \hat{y} \rangle - \langle \hat{y}^T \rangle \langle \hat{y} \rangle = \text{tr} \langle \hat{y}, \hat{y}^T \rangle \quad (132b)$$

$$SS_{res} = \langle \| y - \hat{y} \|^2 \rangle = \langle \| \hat{\epsilon} \|^2 \rangle = \text{tr} \langle \hat{\epsilon}, \hat{\epsilon}^T \rangle \quad (132c)$$

⁴I find the word “homoscedasticity” unnecessarily long, cryptic and easy to misspell so I like to replace it by “homo-spread”. The opposite of “homoscedasticity” is “heteroscedasticity”, which I like to replace with “hetero-spread”.

Claim 8 *The following is true without homo-spread:*

$$\underbrace{\text{tr} \langle \underline{y}, \underline{y}^T \rangle}_{SS_{\underline{y}}} = \underbrace{\text{tr} \langle \hat{\underline{y}}, \hat{\underline{y}}^T \rangle}_{SS_{\hat{\underline{y}}}} + \underbrace{\text{tr} \langle \hat{\underline{\epsilon}}, \hat{\underline{\epsilon}}^T \rangle}_{SS_{res}} \quad (133)$$

This is like the Pythagorean Theorem for the magenta right triangle in Fig.8.

proof:

From Eqs.129 and 132, we see that

$$SS_{\underline{y}} = \text{tr} \langle \underline{y}, \underline{y}^T \rangle \quad (134)$$

$$SS_{\hat{\underline{y}}} = \text{tr} \langle \hat{\underline{y}}, \hat{\underline{y}}^T \rangle = \text{tr} \langle \wedge \underline{y}, \underline{y}^T \rangle \quad (135)$$

$$SS_{res} = \text{tr} \langle \hat{\underline{\epsilon}}, \hat{\underline{\epsilon}}^T \rangle = \text{tr} \langle \vee \underline{y}, \underline{y}^T \rangle \quad (136)$$

Now use $\wedge + \vee = 1$.

QED

The goodness of fit for this model is often measured using the **coefficient of determination** R^2 . R^2 is defined by

$$R^2 = 1 - \frac{SS_{res}}{SS_{\underline{y}}} = \frac{SS_{\hat{\underline{y}}}}{SS_{\underline{y}}} = \frac{\text{tr} \langle \hat{\underline{y}}, \hat{\underline{y}}^T \rangle}{\text{tr} \langle \underline{y}, \underline{y}^T \rangle} \quad (137)$$

If homo-spread holds, then R^2 reduces to

$$R^2 = \frac{\text{tr} \wedge}{nsam} . \quad (138)$$

0.26.2 LR, assuming x_σ are random

Let

$i_0 \in \{0, 1, 2, \dots, n\}$: index that can assume values 0 to n

$i \in \{1, 2, \dots, n\}$: index that can assume values 1 to n . i is never equal to 0.

$\underline{y} \in \mathbb{R}$: true value of dependent y-variable

$\hat{\underline{y}} \in \mathbb{R}$: estimator of dependent y-variable

$\underline{\epsilon} \in \mathbb{R}$: residual

$\underline{x}_i \in \mathbb{R}$: independent x-variables for $i \in \{1, \dots, n\}$

$\beta_0, \beta_i \in \mathbb{R}$: regression coefficients

$$\hat{\underline{y}} = \beta_0 + \sum_{j=1}^n \beta_j \underline{x}_j \quad (139)$$

$$= \sum_{j_0=0}^n \beta_{j_0} \underline{x}_{j_0} \quad (\text{Assume } \underline{x}_0 = 1.) \quad (140)$$

$$\underline{y} = \hat{\underline{y}} + \epsilon \quad (141)$$

Transforming expressions from non-random to random x_σ

Define the following population averages:

$$E_\sigma[x^\sigma] = \frac{1}{nsam} \sum_\sigma x^\sigma, \quad (142)$$

$$E_\sigma[x^\sigma y^\sigma] = \frac{1}{nsam} \sum_\sigma x^\sigma y^\sigma, \quad (143)$$

$$\langle x^\sigma, y^\sigma \rangle_\sigma = E_\sigma[x^\sigma y^\sigma] - E_\sigma[x^\sigma] E_\sigma[y^\sigma]. \quad (144)$$

Claim 9 If the x_σ are i.i.d. random variables,

$$E_\sigma[x^\sigma] = \langle \underline{x} \rangle \quad (145)$$

$$E_\sigma[x^\sigma y^\sigma] = \langle \underline{x} \underline{y} \rangle \quad (146)$$

$$\langle x^\sigma, y^\sigma \rangle_\sigma = \langle \underline{x}, \underline{y} \rangle \quad (147)$$

proof:

$$\frac{1}{nsam} \sum_\sigma x^\sigma = \frac{1}{nsam} \sum_{x \in S_{\underline{x}}} x \underbrace{\sum_\sigma \mathbb{1}(x^\sigma = x)}_{N(x^\sigma = x)} \quad (148)$$

$$= \sum_x x P(x) \quad (149)$$

$$= \langle \underline{x} \rangle \quad (150)$$

$$\frac{1}{nsam} \sum_\sigma x^\sigma y^\sigma = \frac{1}{nsam} \sum_{x \in S_{\underline{x}}} \sum_{y \in S_{\underline{y}}} xy \underbrace{\sum_\sigma \mathbb{1}(x^\sigma = x, y^\sigma = y)}_{N(x^\sigma = x, y^\sigma = y)} \quad (151)$$

$$= \sum_{x,y} xy P(x,y) \quad (152)$$

$$= \langle \underline{x} \underline{y} \rangle \quad (153)$$

Eq.(147) follows from Eq.(145) and Eq.(146).

QED

Recall that

$$Y_\sigma = \beta_0 + \sum_{j=1}^n X_{\sigma,j} \beta_j + \epsilon_\sigma . \quad (154)$$

Assume

$$E_\sigma[X_{\sigma,k} \epsilon_\sigma] = E_\sigma[X_{\sigma,k}] \underbrace{E_\sigma[\epsilon_\sigma]}_{=0} = 0 . \quad (155)$$

Then we have

$$E_\sigma[X_{\sigma,k} Y_\sigma] = E_\sigma[X_{\sigma,k}] \beta_0 + \sum_{j=1}^n E_\sigma[X_{\sigma,k} X_{\sigma,j}] \beta_j + \underbrace{E_\sigma[X_{\sigma,k} \epsilon_\sigma]}_{=0} \quad (156)$$

and

$$E_{\sigma'}[X_{\sigma',k}] E_\sigma[Y_\sigma] = E_{\sigma'}[X_{\sigma',k}] \beta_0 + \sum_{j=1}^n E_{\sigma'}[X_{\sigma',k}] E_\sigma[X_{\sigma,j}] \beta_j + \underbrace{E_{\sigma'}[X_{\sigma',k}] E_\sigma[\epsilon_\sigma]}_{=0} . \quad (157)$$

Subtracting Eq.(157) from Eq.(156), we get

$$\langle X_{\sigma,k}, Y_\sigma \rangle_\sigma = \sum_{j=1}^n \langle X_{\sigma,k}, X_{\sigma,j} \rangle_\sigma \beta_j . \quad (158)$$

Define the n dimensional covariance matrix C by

$$C_{k,j} = \langle X_{\sigma,k}, X_{\sigma,j} \rangle_\sigma . \quad (159)$$

Then Eq.(158) implies

$$\beta_j = \sum_{k=1}^n C_{j,k}^{-1} \langle X_{\sigma,k}, Y_\sigma \rangle_\sigma \quad (160)$$

for all $j = 1, 2, \dots, n$.

If we assume that the x_σ are i.i.d., then, by virtue of Claim 9, the matrix C tends to

$$C_{k,j} \rightarrow \langle \underline{x}_k, \underline{x}_j \rangle \quad (161)$$

and Eq.(160) implies

$$\beta_j = \sum_{k=1}^n C_{j,k}^{-1} \langle \underline{x}_k, \underline{y} \rangle . \quad (162)$$

Geometry of LR with random x_σ

Recall that

$$\underline{y} = \beta_0 + \underbrace{\sum_{j=1}^n \beta_j \underline{x}_j}_{\hat{\underline{y}}} + \underline{\epsilon}. \quad (163)$$

Assume

$$\langle \underline{\epsilon} \rangle = 0 \quad (164)$$

and

$$\langle \underline{x}_j, \underline{\epsilon} \rangle = 0 \quad (165)$$

for all j .

For $k = 1, \dots, n$,

$$\langle \underline{x}_k, \underline{y} \rangle = \sum_{j=1}^n \beta_j \langle \underline{x}_k, \underline{x}_j \rangle. \quad (166)$$

Let \underline{x}^n and β^n be n -dimensional column vectors. Then Eq.(166) can be represented in matrix notation by

$$\langle \underline{x}^n, \underline{y} \rangle = \langle \underline{x}^n, (\underline{x}^n)^T \rangle \beta^n. \quad (167)$$

Hence,

$$\boxed{\beta^n = \langle \underline{x}^n, (\underline{x}^n)^T \rangle^{-1} \langle \underline{x}^n, \underline{y} \rangle.} \quad (168)$$

For β_0 , use

$$\beta_0 = \langle \underline{y} \rangle - \langle \underline{x}^n \rangle^T \beta^n \quad (169)$$

Notice that Eq.(168) for the regression coefficients is the same as Eq.(162). So we have rederived the same formula via a different method.

Next, we will write Eq.(168) for the special cases $n = 1$ and $n = 2$, where n is the number of independent x-variables \underline{x}_j

1. $n = 1$ (y fitted by a line)

$$\underline{y} = \beta_0 + \beta_1 \underline{x} + \underline{\epsilon} \quad (170)$$

Eq.(168) becomes

$$\beta_1 = \frac{\langle \underline{y}, \underline{x} \rangle}{\langle \underline{x}, \underline{x} \rangle} \quad (171)$$

2. $n = 2$ (\underline{y} fitted by a plane)

$$\underline{y} = \beta_0 + \beta_1 \underline{x}_1 + \beta_2 \underline{x}_2 + \epsilon \quad (172)$$

Define

$$C_{i,j} = \langle \underline{x}_i, \underline{x}_j \rangle \quad (173)$$

for all i, j . Then Eq.(168) becomes⁵

$$\begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} = C^{-1} \begin{bmatrix} \langle \underline{y}, \underline{x}_1 \rangle \\ \langle \underline{y}, \underline{x}_2 \rangle \end{bmatrix} \quad (174)$$

$$= \frac{1}{\det C} \begin{bmatrix} C_{22} & -C_{12} \\ -C_{21} & C_{11} \end{bmatrix} \begin{bmatrix} \langle \underline{y}, \underline{x}_1 \rangle \\ \langle \underline{y}, \underline{x}_2 \rangle \end{bmatrix} \quad (175)$$

Hence,

$$\beta_1 = \frac{\partial \underline{y}}{\partial \underline{x}_1} = \frac{C_{22} \langle \underline{y}, \underline{x}_1 \rangle - C_{12} \langle \underline{y}, \underline{x}_2 \rangle}{C_{11} C_{22} - C_{12}^2} \quad (176)$$

Eq.(176) agrees with the value of $\beta_{YX,Z}$ in Ref.[33] by Pearl, if we replace in Pearl's formulae $X \rightarrow \underline{x}_1$, $Y \rightarrow \underline{y}$, $Z \rightarrow \underline{x}_2$.

Regression interpreted as differentiation of y

Finding the derivative of y with respect to (wrt) X (aka “Regressing y on X ”) is finding $\hat{\beta} = \frac{d}{dX} \underbrace{\underline{y}}_{X\beta}$.

Recall that

$$\underline{y} = \beta_0 + \sum_{k=1}^n \underline{x}_k \beta_k + \epsilon. \quad (177)$$

Therefore,

$$\langle \underline{x}_i, \underline{y} \rangle = \sum_{k=1}^n \langle \underline{x}_i, \underline{x}_k \rangle \beta_k \quad (178)$$

$$= \langle \underline{x}_i, \underline{x}_i \rangle \beta_i + \sum_{k=1}^n \mathbb{1}(k \neq i) \langle \underline{x}_i, \underline{x}_k \rangle \beta_k. \quad (179)$$

⁵Recall that if $M = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ then $M^{-1} = \frac{1}{\det M} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$

Hence,

$$\beta_i = \frac{\langle \underline{x}_i, \underline{y} \rangle}{\langle \underline{x}_i, \underline{x}_i \rangle} - \sum_{k=1}^n \mathbb{1}(k \neq i) \frac{\langle \underline{x}_i, \underline{x}_k \rangle}{\langle \underline{x}_i, \underline{x}_i \rangle} \beta_k . \quad (180)$$

Let's represent the linear operator $\langle \underline{x}_i, \cdot \rangle^{-1} \langle \underline{x}_i, \cdot \rangle$ as a derivative:

$$\frac{d \cdot}{d \underline{x}_i} = \frac{\langle \underline{x}_i, \cdot \rangle}{\langle \underline{x}_i, \underline{x}_i \rangle} . \quad (181)$$

Eq.(180) can be expressed in derivative notation as:

$$\beta_i = \frac{d \underline{y}}{d \underline{x}_i} - \sum_{k=1}^n \mathbb{1}(k \neq i) \frac{d \underline{x}_k}{d \underline{x}_i} \beta_k \quad (182)$$

Note that Eq.(182) evokes the formula for differentials:

$$d \underline{y} = \sum_{k=1}^n \beta_k d \underline{x}_k . \quad (183)$$

Note also that, because of the linearity of the derivative operator, Eq.(182) implies:

$$\beta_i = \frac{d}{d \underline{x}_i} \left(\underbrace{\underline{y} - \sum_{k=1}^n \mathbb{1}(k \neq i) \underline{x}_k \beta_k}_{\underline{y} - \underline{x}_i \beta_i} \right) . \quad (184)$$

Eq.(184) can be used to find $\hat{\beta}_i$ in two steps:

STEP 1: Regress $\underline{y} - \underline{x}_i \beta_i$ on $(\underline{x}_k)_{k \in \{1, 2, \dots, n\} - \{i\}}$. Get estimates $(\hat{\beta}_k)_{k \in \{1, 2, \dots, n\} - \{i\}}$.

STEP 2: Regress $\underline{y} - \sum_{k \neq i} \underline{x}_k \hat{\beta}_k$ on \underline{x}_i . Get estimate $\hat{\beta}_i$.

Of course, one can also find $\hat{\beta}_i$ by regressing \underline{y} on $(\underline{x}_k)_{k \in \{1, 2, \dots, n\}}$, to get estimates $(\hat{\beta}_k)_{k \in \{1, 2, \dots, n\}}$.

0.27 Logistic Regression (LoR)

Suppose $x_\sigma \in \mathbb{R}^n$, $y_\sigma \in \mathbb{R}$, and Σ is a population of individuals σ . In general, a **regression** is when we curve-fit a dataset $\{(x_\sigma, y_\sigma) : \sigma \in \Sigma\}$ with a function $\hat{y} = f(x)$. In **Linear Regression (LR)**, which we discussed earlier, $f(x)$ is a hyperplane in x . On the other hand, in **Logistic Regression (LoR)**, $y_\sigma \in [0, 1]$ and $f(x)$ is the sigmoid of a hyperplane in x .

More specifically, for LR we have Eq.(111) which reads as follows:

$$y_\sigma = \beta_0 + \underbrace{\sum_{i=1}^n x_{\sigma i} \beta_i}_{\hat{y}_\sigma} + \epsilon_\sigma \quad (\text{LR}) . \quad (185)$$

For LoR, we have instead

$$p_\sigma = \text{smoid} \left(\beta_0 + \sum_{i=1}^n x_{\sigma i} \beta_i + \epsilon_\sigma \right) \quad (\text{LoR}) , \quad (186)$$

or, equivalently,

$$\underbrace{\text{logds}(p_\sigma)}_{\ln \frac{p_\sigma}{1-p_\sigma}} = \beta_0 + \underbrace{\sum_{i=1}^n x_{\sigma i} \beta_i}_{\hat{y}_\sigma} + \epsilon_\sigma \quad (\text{LoR}) , \quad (187)$$

where we have used the fact that $\text{logds}()$ is the inverse function of $\text{smoid}()$. Hence, an LoR fit can be calculated by collecting a dataset $\{(x_\sigma, p_\sigma) : \sigma \in \Sigma\}$, transforming that dataset to the dataset $\{(x_\sigma, \text{logds}(p_\sigma)) : \sigma \in \Sigma\}$, and fitting the latter dataset with a hyperplane. Let $P(\underline{Y}_\sigma = 1) = p_\sigma \in [0, 1]$. LoR can be used for binary classification if we define the binary class variable $c_\sigma \in \{0, 1\}$ by

$$c_\sigma = \mathbb{1}(P(\underline{Y}_\sigma = 1) > \alpha) \quad (188)$$

for some $0 < \alpha < 1$.

0.28 Entropy, Kullback-Liebler divergence, Cross-Entropy

For probability distributions $p(x), q(x)$ of $x \in S_x$

- Entropy:

$$H(p) = - \sum_x p(x) \ln p(x) \geq 0 \quad (189)$$

- Kullback-Liebler divergence:

$$D_{KL}(p \parallel q) = \sum_x p(x) \ln \frac{p(x)}{q(x)} \geq 0 \quad (190)$$

- Cross entropy:

$$CE(p \parallel q) = - \sum_x p(x) \ln q(x) \quad (191)$$

$$= H(p) + D_{KL}(p \parallel q) \quad (192)$$

0.29 Definition of various entropies used in Shannon Information Theory

- (plain) Entropy of \underline{x}

$$H(\underline{x}) = - \sum_x P(x) \ln P(x) \quad (193)$$

This quantity measures the spread of $P_{\underline{x}}$. $H(\underline{x}) \geq 0$ and it vanishes iff $P(x) = \delta(x, x_0)$ (deterministic case)

- Conditional Entropy of \underline{y} given \underline{x}

$$H(\underline{y}|\underline{x}) = - \sum_{x,y} P(x,y) \ln P(y|x) \quad (194)$$

$$= H(\underline{y}, \underline{x}) - H(\underline{x}) \quad (195)$$

This quantity measures the conditional spread of \underline{y} given \underline{x} . $H(\underline{y}|\underline{x}) \geq 0$.

- Mutual Information (MI) of \underline{x} and \underline{y} .

$$H(\underline{y} : \underline{x}) = \sum_{x,y} P(x,y) \ln \frac{P(x,y)}{P(x)P(y)} \quad (196)$$

$$= H(\underline{x}) + H(\underline{y}) - H(\underline{y}, \underline{x}) \quad (197)$$

This quantity measures the correlation between \underline{x} and \underline{y} . $H(\underline{y} : \underline{x}) \geq 0$ and it vanishes iff $P(x,y) = P(x)P(y)$.

- Conditional Mutual Information (CMI)⁶ of \underline{x} and \underline{y} given $\underline{\lambda}$

$$H(\underline{y} : \underline{x} | \underline{\lambda}) = \sum_{x,y,\lambda} P(x,y,\lambda) \ln \frac{P(x,y|\lambda)}{P(x|\lambda)P(y|\lambda)} \quad (198)$$

$$= H(\underline{x} | \underline{\lambda}) + H(\underline{y} | \underline{\lambda}) - H(\underline{y}, \underline{x} | \underline{\lambda}) \quad (199)$$

This quantity measures the conditional correlation of \underline{x} and \underline{y} given $\underline{\lambda}$. $H(\underline{y} : \underline{x} | \underline{\lambda}) \geq 0$ and it vanishes iff $P(x,y|\lambda) = P(x|\lambda)P(y|\lambda)$.

An interesting special case occurs when $P(\lambda) = \delta(\lambda, \lambda_0)$ (the frequentist case of no λ prior.) In that case CMI reduces to

⁶CMI can be read as “see me”.

$$H(\underline{y} : \underline{x} | \lambda_0) = \sum_{x,y} P(x,y|\lambda_0) \ln \frac{P(x,y|\lambda_0)}{P(x|\lambda_0)P(y|\lambda_0)} \geq 0 \quad (200)$$

- **Kullback-Liebler Divergence from $P_{\underline{x}}$ to $P_{\underline{y}}$.**

Assume random variables \underline{x} and \underline{y} have the same set of states $S_{\underline{x}} = S_{\underline{y}}$. Then

$$D_{KL}(P_{\underline{x}} \| P_{\underline{y}}) = \sum_x P_{\underline{x}}(x) \ln \frac{P_{\underline{x}}(x)}{P_{\underline{y}}(x)} \quad (201)$$

This quantity measures a non-symmetric distance between the probability distributions $P_{\underline{x}}$ and $P_{\underline{y}}$. $D_{KL}(P_{\underline{x}} \| P_{\underline{y}}) \geq 0$ and it equals zero iff $P_{\underline{x}} = P_{\underline{y}}$.

0.30 Arc Strength (Arc Force)

Given a bnet with an arc (i.e., arrow) $\underline{x} \rightarrow \underline{y}$, we define the **arc strength or arc force** of arc $\underline{x} \rightarrow \underline{y}$ to be $H(\underline{x} : \underline{y})$ (i.e., the mutual information between \underline{x} and \underline{y}). Evaluation of $H(\underline{x} : \underline{y})$ requires knowing $P(y|x)$, $P(x)$ and $P(y)$. $P(y|x)$ is the TPM of node \underline{y} , so it is immediately available from the specification of the bnet. Calculating $P(x)$ and $P(y)$ is more involved, and requires marginalizing the full probability distribution of the bnet. Such marginalizations can be done using the junction tree algorithm described in Chapter 35.

0.31 Pearson Chi-Squared Test

The **Pearson divergence** (aka **Pearson Chi-squared test statistic**) for two probability distributions $PO(x)$ and $PE(x)$, where $x \in S_{\underline{x}}$, is defined as follows:

$$D_{\chi^2} = \sum_x \frac{[PO(x) - PE(x)]^2}{PE(x)} = \sum_x \frac{PO^2(x)}{PE(x)} - 1 . \quad (202)$$

Usually PO is the observed probability distribution and PE is the expected, theoretical one.

As the following claim shows, the Pearson divergence is closely related to the Kullback-Liebler divergence.

Claim 10 If $\left| \frac{PO(x)}{PE(x)} - 1 \right| << 1$ for all $x \in S_{\underline{x}}$, then

$$D_{KL}(PO \| PE) \approx D_{\chi^2} . \quad (203)$$

proof:

$$D_{KL}(PO \parallel PE) = \sum_x PO(x) \ln \frac{PO(x)}{PE(x)} \quad (204)$$

$$= \sum_x PO(x) \ln \left(1 + \frac{PO(x)}{PE(x)} - 1 \right) \quad (205)$$

$$\approx \sum_x PO(x) \left(\frac{PO(x)}{PE(x)} - 1 \right) \quad (206)$$

$$= \sum_x \frac{PO^2(x)}{PE(x)} - 1 \quad (207)$$

$$= D_{\chi^2} \quad (208)$$

QED

Let $nx = |S_{\underline{x}}|$. Let $P_{\chi^2}(y)$ be the χ^2 (with $nx - 1$ degrees of freedom) probability distribution, and let $F_{\chi^2}(\alpha)$ be its cumulative distribution. Find α such that

$$95\% = \int_0^\alpha dy P_{\chi^2}(y) = F_{\chi^2}(\alpha) \quad (209)$$

If $D_{\chi^2} < \alpha$, then we say that $PO = PE$ to 95% significance level (SL), whereas if $D_{\chi^2} > \alpha$, we say that $PO \neq PE$ to 95% SL (i.e., SL = 95%). The higher SL becomes, the higher α becomes, and the bigger the divergence D_{χ^2} has to be, before we are willing to declare that $PO \neq PE$.

0.32 Demystifying Population and Sample Variances

Let $x[\sigma] = x^\sigma$. Given i.i.d.real variables $(x^\sigma)_{\sigma=0,1,\dots,n-1}$, let⁷

$$\hat{\mu} = \bar{x} = \frac{1}{n} \sum_{\sigma} x^\sigma \quad (210)$$

$$(\widehat{\sigma^2})_\infty = \frac{1}{n} \sum_{\sigma} (x^\sigma - \mu)^2 \quad (211)$$

$$\widehat{\sigma^2} = \frac{1}{n-1} \sum_{\sigma} (x^\sigma - \hat{\mu})^2 \quad (212)$$

⁷Do not confuse the sample index σ and the standard deviation σ .

Statisticians⁸ call $(\hat{\sigma}^2)_\infty$ the “population variance”. I will call it the **population variance for fixed μ** . Note that it depends on the fixed parameter μ . Statisticians call $\hat{\sigma}^2$ the “sample variance”. Instead, I will call $\hat{\sigma}^2$ the **population variance for random μ** .

If one treats \underline{x}^σ as a random variable, then one must treat $\hat{\mu}$ as a random variable too. Let

$$E[\underline{x}^\sigma] = \mu \quad (213)$$

and

$$\langle \underline{x}^\sigma, \underline{x}^{\sigma'} \rangle = \delta(\sigma, \sigma') \sigma^2. \quad (214)$$

Then one can show that

$$E[(\hat{\sigma}^2)_\infty] = \frac{1}{n} E \left[\sum_{\sigma} (\underline{x}^\sigma - \mu)^2 \right] \quad (215)$$

$$= \sigma^2 \quad (216)$$

and

$$E[\hat{\sigma}^2] = \frac{1}{n-1} E \left[\sum_{\sigma} (\underline{x}^\sigma - \hat{\mu})^2 \right] \quad (217)$$

$$= \sigma^2 \quad (218)$$

This is the reason why we use an $n - 1$ instead of an n in $\hat{\sigma}^2$. Because it makes $E[\hat{\sigma}^2] = \sigma^2$ so $\hat{\sigma}^2$ is an unbiased estimator of the single individual variance σ^2 .

The intuitive reason for why $\hat{\sigma}^2$ is divided by $n - 1$ instead of n is that whereas μ in $(\hat{\sigma}^2)_\infty$ is kept fixed and is “quiet”, the $\hat{\mu}$ in $\hat{\sigma}^2$ is a random variable, noisy instead of quiet. The fluctuations in $\hat{\mu}$ are strongly correlated with the fluctuations of the \underline{x}^σ , so they decrease the fluctuations in $\hat{\sigma}^2$ compared to those in $(\hat{\sigma}^2)_\infty$. By dividing by $n - 1$ instead of n , we compensate for this decrease in fluctuations so that the ratio of the numerator and denominator of $\hat{\sigma}^2$ equals σ^2 , instead of something *smaller* than σ^2 , as would happen if were to divide by n instead of $n - 1$. In terms of “degrees of freedom”(DOFs), $(\hat{\sigma}^2)_\infty$ has n DOFs (namely one for each \underline{x}^σ), whereas $\hat{\sigma}^2$ has $n - 1$ DOFs. (the presence of $\hat{\mu}$ subtracts one DOF). In both $(\hat{\sigma}^2)_\infty$ and $\hat{\sigma}^2$, one divides by the number of DOFs.

⁸In the language of Statisticians, a “population” is supposed to be so large that its μ does not fluctuate, and a “sample” is supposed to be a small subset of that population for which the μ is assumed to fluctuate. In this book, I use the word “population” to mean a set of any size containing individuals, I use the word “sub-population” to refer to a subset of the population, and I use the word “sample” (aka individual, observation, unit, record) to mean a single individual of the population.

0.33 Independence of $\hat{\mu}$ and $\hat{\sigma}^2$

Let $x[\sigma] = x^\sigma$. Consider i.i.d.real variables $(x^\sigma)_{\sigma=0,1,\dots,n-1}$ such that⁹

$$E[\underline{x}^\sigma] = \mu \quad (219)$$

$$\langle \underline{x}^\sigma, \underline{x}^{\sigma'} \rangle = \delta(\sigma, \sigma') \sigma^2 . \quad (220)$$

$$\hat{\mu} = \bar{x} = \frac{1}{n} \sum_{\sigma} x^\sigma \quad (221)$$

$$(\hat{\sigma}^2)_\infty = \frac{1}{n} \sum_{\sigma} (x^\sigma - \mu)^2 \quad (222)$$

$$\hat{\sigma}^2 = \frac{1}{n-1} \sum_{\sigma} (x^\sigma - \hat{\mu})^2 \quad (223)$$

Claim 11 Let

$$\underline{\Delta}^\sigma = \underline{x}^\sigma - \mu . \quad (224)$$

For any $\sigma_1, \sigma_2, \sigma_3$,

$$\langle \underline{\Delta}^{\sigma_1} \underline{\Delta}^{\sigma_2}, \underline{\Delta}^{\sigma_3} \rangle = 0 . \quad (225)$$

proof:

Suppose $\sigma_2 \neq \sigma_3$. Then

$$\langle \underline{\Delta}^{\sigma_1} \underline{\Delta}^{\sigma_2}, \underline{\Delta}^{\sigma_3} \rangle = \underbrace{\langle \underline{\Delta}^{\sigma_2} \rangle}_0 \langle \underline{\Delta}^{\sigma_1}, \underline{\Delta}^{\sigma_3} \rangle = 0 . \quad (226)$$

So assume $\sigma_2 = \sigma_3 = \sigma$ and evaluate $\langle \underline{\Delta}^{\sigma_1} \underline{\Delta}^\sigma, \underline{\Delta}^\sigma \rangle$.

Suppose $\sigma_1 \neq \sigma$. Then

$$\langle \underline{\Delta}^{\sigma_1} \underline{\Delta}^\sigma, \underline{\Delta}^\sigma \rangle = \underbrace{\langle \underline{\Delta}^{\sigma_1} \rangle}_0 \langle \underline{\Delta}^\sigma, \underline{\Delta}^\sigma \rangle = 0 . \quad (227)$$

So suppose $\sigma_1 = \sigma$ and evaluate $\langle (\underline{\Delta}^\sigma)^2, \underline{\Delta}^\sigma \rangle$.

$$\langle (\underline{\Delta}^\sigma)^2, \underline{\Delta}^\sigma \rangle = \underbrace{\langle (\underline{\Delta}^\sigma)^3 \rangle}_0 - \underbrace{\langle (\underline{\Delta}^\sigma)^2 \rangle}_0 \underbrace{\langle \underline{\Delta}^\sigma \rangle}_0 = 0 . \quad (228)$$

QED

⁹Do not confuse the sample index σ and the standard deviation σ .

Claim 12

$$\left\langle \widehat{\underline{\sigma^2}}, \widehat{\underline{\mu}} \right\rangle = 0 . \quad (229)$$

proof:

$$\left\langle \widehat{\underline{\sigma^2}}, \widehat{\underline{\mu}} \right\rangle = \frac{1}{n(n-1)} \sum_{\sigma, \sigma'} \left\langle \left(\underline{x}^\sigma - \frac{1}{n} \sum_{\sigma''} \underline{x}^{\sigma''} \right)^2, \underline{x}^{\sigma'} \right\rangle \quad (230)$$

$$= \frac{1}{n(n-1)} \sum_{\sigma, \sigma'} \left\langle \left(\underline{x}^\sigma - \frac{1}{n} \sum_{\sigma''} \underline{x}^{\sigma''} \right)^2, \underline{\Delta}^{\sigma'} \right\rangle \quad (231)$$

$$= \frac{1}{n(n-1)} \sum_{\sigma, \sigma'} \left\langle \left(\underline{\Delta}^\sigma - \frac{1}{n} \sum_{\sigma''} \underline{\Delta}^{\sigma''} \right)^2, \underline{\Delta}^{\sigma'} \right\rangle \quad (232)$$

$$= 0 \quad \text{by Claim 11 .} \quad (233)$$

QED

0.34 Chi-square distribution

This section is based on Ref.[78].

$$\underline{q} \longleftarrow \underline{z}.$$

Figure 9: Bnet used to define the Chi-square distribution.

Let $q \in \mathbb{R}$ and $\underline{z} = \{z_i\}_{i=0,1,\dots,\nu-1}$ where $z_i \in \mathbb{R}$. Consider the bnet of Fig.9. The TPMs, printed in blue, for that bnet, are as follows:¹⁰

$$P(z_i) = \mathcal{N}(z_i; \mu = 0, \sigma^2 = 1) . \quad (234)$$

We want

$$\underline{q} = \sum_{i=0}^{\nu-1} (\underline{z}_i)^2 \quad (235)$$

so $P(q|\underline{z})$ is a Dirac delta function:

$$P(q|\underline{z}) = \delta(q - \sum_{i=0}^{\nu-1} (z_i)^2) . \quad (236)$$

¹⁰Don't confuse the q independent constant $\mathcal{N}(!q)$ with the normal probability distribution $\mathcal{N}(x; \mu, \sigma^2)$.

Therefore

$$P(q) = \prod_{i=0}^{\nu-1} \left\{ \int dz_i P(z_i) \right\} P(q|z.) \quad (237)$$

$$= \mathcal{N}(!q) q^{\frac{\nu}{2}-1} e^{-q/2} = \chi^2(q; \nu), \quad (238)$$

where $\mathcal{N}(!q)$ is a constant that does not depend on q and is adjusted so that $\int_0^\infty dq P(q) = 1$.

0.35 Student's t-distribution

This section is based on Ref.[127].

Let $x[\sigma] = x^\sigma$. Consider i.i.d.real variables $(x^\sigma)_{\sigma=0,1,\dots,n-1}$ such that¹¹

$$E[\underline{x}^\sigma] = \mu \quad (239)$$

$$\langle \underline{x}^\sigma, \underline{x}^{\sigma'} \rangle = \delta(\sigma, \sigma') \sigma^2. \quad (240)$$

$$\hat{\mu} = \bar{x} = \frac{1}{n} \sum_{\sigma} x^\sigma \quad (241)$$

$$(\widehat{\sigma^2})_\infty = \frac{1}{n} \sum_{\sigma} (x^\sigma - \mu)^2 \quad (242)$$

$$\widehat{\sigma^2} = \frac{1}{n-1} \sum_{\sigma} (x^\sigma - \hat{\mu})^2. \quad (243)$$

If we define

$$z = \frac{\hat{\mu} - \mu}{\frac{\sigma}{\sqrt{n}}}, \quad (244)$$

then \underline{z} has a Standard Normal Distribution (SND):

$$P(z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} = \mathcal{N}(z; \mu = 0, \sigma^2 = 1) \quad (245)$$

But what if we allow the standard deviation σ to fluctuate in the expression Eq.(244) for z ? Define

$$t = \frac{\hat{\mu} - \mu}{\sqrt{\frac{\widehat{\sigma^2}}{n}}}. \quad (246)$$

¹¹Do not confuse the sample index σ and the standard deviation σ .

Then one can show that \underline{t} has the **Student's t-distribution** $\text{Stud}(t; \nu = n - 1)$ given by:

$$P(t) = \mathcal{N}(!t)(1 + \frac{t^2}{\nu})^{-\frac{\nu+1}{2}} = \text{Stud}(t; \nu = n - 1) \quad (247)$$

Note that if we use the approximation $e^x \approx 1 + x + \mathcal{O}(x^2)$, we can show that $\text{Stud}(t)$ tends to the SND when $n \gg 1$:

$$P(t) = \mathcal{N}(!t)(1 + \frac{t^2}{\nu})^{-\frac{\nu+1}{2}} \quad (248)$$

$$\approx \mathcal{N}(!t)e^{-\frac{t^2}{2}\frac{\nu+1}{\nu}} \quad (249)$$

$$\approx \mathcal{N}(t; \mu = 0, \sigma^2 = 1). \quad (250)$$

Partial derivation of the explicit form of $\text{Stud}(t)$.

Note that the z definition Eq.(244) and the t definition Eq.(246), imply that

$$t = z \underbrace{\sqrt{\frac{\sigma^2}{\hat{\sigma}^2}}}_{\varrho}, \quad (251)$$

In the expression $\underline{t} = \underline{z}\varrho$, the random variables \underline{z} and ϱ are independent because, as shown in Section 0.33, $\hat{\mu}$ and $\hat{\sigma}^2$ are independent. Therefore, the random variable \underline{t} can be defined using the bnet of Fig.10.

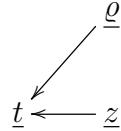


Figure 10: Bnet used to define the Student's t-distribution.

The TPMs, printed in blue, for the bnet Fig.10, are as follows:

$$P(t|z, \varrho) = \delta(t - z\varrho) \quad (\text{Dirac delta function}) \quad (252)$$

$$P(z) = \mathcal{N}(z; \mu = 0, \sigma^2 = 1) \quad (253)$$

$$P(\varrho) = \text{given by Eq.(266) below.} \quad (254)$$

Note that

$$P(\underline{t} = t) = P(\underline{z}\underline{\varrho} = t) \quad (255)$$

$$= \int d\underline{\varrho} P(\underline{z} = \frac{t}{\underline{\varrho}} | \underline{\varrho}) P(\underline{\varrho}) \quad (256)$$

$$= \int d\underline{\varrho} \mathcal{N}(\frac{t}{\underline{\varrho}}; 0, 1) P(\underline{\varrho}) \quad (257)$$

$$= \int d\underline{\varrho} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{t}{\underline{\varrho}})^2} P(\underline{\varrho}) . \quad (258)$$

If we define q by

$$q = \frac{n-1}{\underline{\varrho}^2} , \quad (259)$$

then

$$q = \frac{(n-1)\widehat{\sigma}^2}{\sigma^2} = \frac{1}{\sigma^2} \sum_{\sigma=0}^{n-1} (x^\sigma - \hat{\mu})^2 . \quad (260)$$

As a consequence of ‘‘Cochran’s Theorem’’ (see Ref.[80]), q given by Eq.(260) must have a Chi-square probability distribution with $\nu = n - 1$ degrees of freedom:¹²

$$P(q) = \chi^2(q; \nu = n - 1) \quad (261)$$

Henceforth, let $\nu = n - 1$. From the definition Eq.(259) of q , we get

$$dq = \frac{-2\nu}{\underline{\varrho}^3} d\underline{\varrho} . \quad (262)$$

Therefore,

$$P(\underline{\varrho})d\underline{\varrho} = P(q)dq \quad (263)$$

$$= \chi^2\left(\frac{\nu}{\underline{\varrho}^2}; \nu\right) \frac{(-2\nu)}{\underline{\varrho}^3} d\underline{\varrho} \quad (264)$$

$$= \mathcal{N}(!\underline{\varrho}) \left(\frac{\nu}{\underline{\varrho}^2}\right)^{\frac{\nu}{2}-1} e^{-\frac{\nu}{2\underline{\varrho}^2}} \frac{d\underline{\varrho}}{\underline{\varrho}^3} \quad (265)$$

$$= \mathcal{N}(!\underline{\varrho}) \frac{d\underline{\varrho}}{\underline{\varrho}^{\nu+1}} e^{-\frac{\nu}{2\underline{\varrho}^2}} . \quad (266)$$

¹²Note that this q is a quadratic form $q = \vec{x}^T M \vec{x}$, where \vec{x} is an n dimensional column vector with components x^σ , and M is an $n \times n$ matrix. Cochran’s Theorem diagonalizes M and replaces the vectors \vec{x} by equivalent ones in a new basis. Then the number of *DOFs* (degrees of freedom) of the chi-square distribution is the number of non-zero diagonal elements in the diagonalized M (this number is called the rank of M). In the particular case of Eq.(260), $DOF = n - 1$.

Hence,

$$P(t) = \mathcal{N}(!t) \int_0^\infty \frac{d\varrho}{\varrho^{\nu+1}} e^{-\frac{\nu}{2\varrho^2}} e^{-\frac{1}{2}(\frac{t}{\varrho})^2} \quad (267)$$

$$= \mathcal{N}(!t) \int_0^\infty \frac{d\varrho}{\varrho^{\nu+1}} e^{-\frac{1}{2}\frac{t^2+\nu}{\varrho^2}}. \quad (268)$$

0.36 Hypothesis testing and 3 classic test statistics (Likelihood, Score, Wald)

Suppose we have data $\vec{x} = [x^\sigma]_{\sigma=0,1,\dots,n_{sam}-1}$ which is distributed according to a probability distribution $P(\vec{x}; \theta)$ which depends on some parameters $\theta = (\theta_i)_{i=0,1,\dots,n-1} \in \mathbb{R}^n$. We define the

- **Likelihood of θ** by

$$L(\theta) = P(\vec{x}|\theta), \quad (269)$$

- the **Maximum Likelihood estimate of θ** by

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} L(\theta), \quad (270)$$

- the **Log-likelihood of θ** by

$$LL(\theta) = \ln L(\theta), \quad (271)$$

- the **Score or Lagrange Multiplier vector¹³** of θ by

$$sc(\theta) = [sc_i(\theta)] \in \mathbb{R}^n, \quad (272)$$

where

$$sc_i(\theta) = \partial_{\theta_i} LL(\theta), \quad (273)$$

¹³ $sc_i(\theta) = \partial_{\theta_i} LL(\theta)$ is called a Lagrange multiplier because to maximize $LL(\theta)$ over θ subject to the constraint $\theta = \theta^*$, we maximize the Lagrangian $\mathcal{L} = LL(\theta) - \sum_i \alpha_i [\theta_i - \theta_i^*]$ over (θ_i, α_i) for all i . The latter gives $\alpha_i = \partial_{\theta_i} LL(\theta^*)$ and $\theta = \theta^*$. For a general constrained optimization problem, $\mathcal{L} = LL(\theta) - \sum_i \alpha_i c_i(\theta)$ for some constraint functions $c_i(\theta)$. Hence, $\alpha_i = \partial_{\theta_i} LL / \partial_{\theta_i} c_i$ and $c_i(\theta) = 0$ for all i . So, in general, a Lagrange multiplier is a ratio of two partial derivatives.

- and the **Fisher Information Matrix** of θ by

$$FI(\theta) = [FI_{i,j}(\theta)] \in \mathbb{R}^{n \times n} \quad (274)$$

where

$$FI_{i,j}(\theta) = -E_{\vec{x}|\theta}[\partial_{\theta_i} \partial_{\theta_j} \overbrace{\ln P(\vec{x}|\theta)}^{LL(\theta)}] . \quad (275)$$

Note that if

$$LL(\theta) = \ln P(x|\theta) = \frac{-(\theta - \hat{\theta})^2}{2\sigma^2} + \mathcal{N}(!\theta) , \quad (276)$$

then

$$sc(\theta) = \frac{-(\theta - \hat{\theta})}{\sigma^2} \quad (277)$$

and

$$FI(\theta) = \frac{1}{\sigma^2} \quad (278)$$

In hypothesis testing, one has a **null hypothesis** $H_0: \theta = \theta^*$, and an **alternative hypothesis** $H_1: \theta \neq \theta^*$. We use a **test statistic** $\lambda(\theta^*, \hat{\theta}) \geq 0$ which measures a kind of distance or separation between the estimate $\hat{\theta}$ and the value θ^* for the null Hypothesis H_0 . For some **confidence level** $C > 0$, if $\lambda(\theta^*, \hat{\theta}) > C$, H_0 is **rejected**, whereas if $\lambda(\theta^*, \hat{\theta}) \leq C$, H_0 is **accepted**.

$\alpha = 1 - C$ is called the **significance level**. Usually $\alpha \ll 1$ represents the area under both tails of a Normal distribution, and $C \approx 1$ represents all the area except the tails of a Normal distribution.

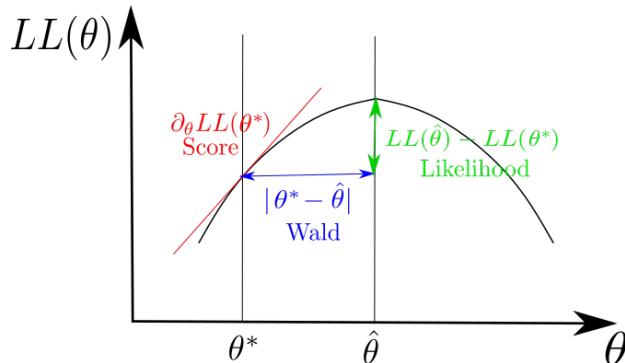


Figure 11: For $n = 1$ and $\theta \in \mathbb{R}$, this figure shows the geometrical significance of certain quantities that characterize the 3 classic test statistics (Likelihood, Score, Wald) for hypothesis testing.

Henceforth in this section, we will occasionally use the Einstein summation convention; i.e., implicit sum over repeated indices.

Three classic test statistics are (See Fig.11):

1. **Likelihood Ratio test statistic** (Ref.[105].)

$$\lambda_{Li} = 2 \ln \left[\frac{L(\hat{\theta})}{L(\theta^*)} \right] = 2[LL(\hat{\theta}) - LL(\theta^*)] \quad (279)$$

2. **Score (a.k.a. Lagrange multiplier) test statistic** (Ref.[123].)

$$\lambda_{Sc} = \partial_{\theta_i} LL(\theta^*) [FI(\theta^*)^{-1}]_{i,j} \partial_{\theta_j} LL(\theta^*) \quad (280)$$

$$= \frac{[\partial_{\theta} LL(\theta^*)]^2}{FI(\theta^*)} \quad \text{if } n = 1 \quad (281)$$

Doesn't depend on $\hat{\theta}$.

3. **Wald test statistic** (Ref.[136].)

$$\lambda_{Wa} = (\hat{\theta} - \theta^*)_i \left[\langle \hat{\theta}, \hat{\theta}^T \rangle^{-1} \right]_{i,j} (\hat{\theta} - \theta^*)_j \quad (282)$$

$$= \frac{(\theta^* - \hat{\theta})^2}{\langle \hat{\theta}, \hat{\theta} \rangle} \quad \text{if } n = 1 \quad (283)$$

More generally, one can replace $\theta^* \rightarrow R\theta^*$ and $\hat{\theta} \rightarrow R\hat{\theta}$ in Eq.(282), where θ^* and $\hat{\theta}$ are n dimensional column vectors, and $R \in \mathbb{R}^{\nu \times n}$. The null and alternative hypotheses become: $H_0 : R\theta = R\theta^*$ and $H_1 : R\theta \neq R\theta^*$. Note that ν is the number of constraints imposed by the null hypothesis. R is called a reparametrization of θ . The Wald test is not reparametrization invariant (i.e., R invariant), but the Likelihood Ratio test is.

Note that if $LL(\theta)$ is given by Eq.(276), then $\langle \hat{\theta}, \hat{\theta} \rangle = \sigma^2 = \frac{1}{FI(\theta)}$. Hence,

$$\lambda_{Li} = \lambda_{Sc} = \lambda_{Wa} = \frac{(\hat{\theta} - \theta^*)^2}{\sigma^2} \quad (284)$$

Many other commonly used test statistics (or their squares) are special cases of one of the 3 classic test statistics. For example, the z-statistic used with normal distributions, the t-statistic used with the Student t-distribution, the F-statistic used in linear regression, the chi-squared statistic used to do Pearson's chi-squared test.

Asymptotic Behavior

If the data \vec{x} is i.i.d.,

$$P(\vec{x}|\theta) = \prod_{\sigma=0}^{nsam-1} P(x^\sigma|\theta) \quad (285)$$

Hence, as $nsam \rightarrow \infty$,

$$LL(\theta) = \ln P(\vec{x}|\theta) \quad (286)$$

$$= \sum_{\sigma} \ln P(x^\sigma|\theta) \quad (287)$$

$$\rightarrow nsam \sum_x P(x|\theta) \ln P(x|\theta) \quad (288)$$

$$= -nsam H(\underline{x}|\theta) \quad (289)$$

Thus, *maximizing* the log likelihood $LL(\theta)$ and *minimizing* the entropy $H(\underline{x}|\theta)$ give the same estimate $\hat{\theta}$.

When the data is i.i.d. and $nsam \rightarrow \infty$, it is also possible to prove that the 3 test statistics defined above all tend to the same probability distribution, namely $\mathcal{X}^2(\theta^*; \nu)$, the chi-square distribution with ν degrees of freedom, where $\theta \in \mathbb{R}^n$, $R \in \mathbb{R}^{\nu \times n}$, and $\nu = n$ if $R = 1$.

0.37 Error Bars

Never report measurements without error bars!!

Assume a distribution with mean μ and standard deviation σ for a subpopulation with n samples.

$SE = \frac{\sigma}{\sqrt{n}}$ is called the **standard error**.

Some popular types of error bars:

- **Box and Whiskers plot (aka Boxplot)**

See Fig.12. IQR stands for **Intermediate Quantile Range**. Sometimes, the endpoints of the error bars are taken to be the minimum and maximum samples instead of $Q_1 - 1.5 * IQR$ and $Q_3 + 1.5 * IQR$. The points that fall in the intervals $[\min, Q_1 - 1.5 * IQR]$ and $[Q_3 + 1.5 * IQR, \max]$ are called **outliers**.

- **Standard Deviation**

Error bar endpoints are located one standard deviation away from the mean.

$$\mu - \sigma < \mu < \mu + \sigma \quad (290)$$

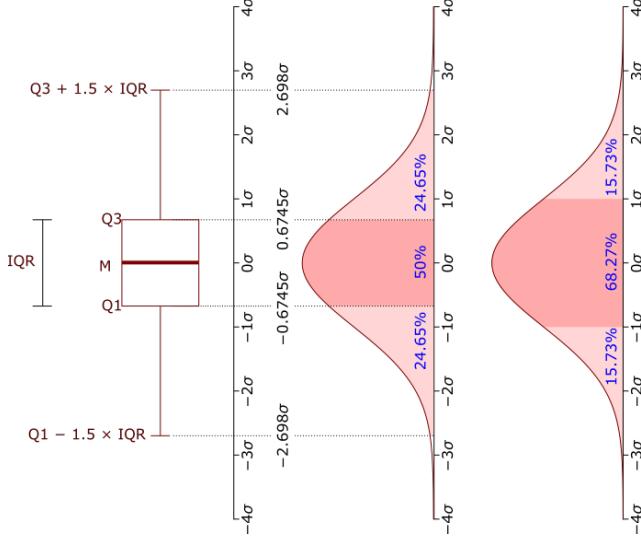


Figure 12: Boxplot plot for Normal distribution $\mathcal{N}(\mu = 0, \sigma)$. Q_1 and Q_3 are the first and third quantiles, and M is the median (i.e., half-way point). For a non-normal skewed distribution, Q_1 and Q_3 are not equidistant from the median, and the median is not exactly equal to the mean.

- **Confidence Interval**

$$\mu - |z^*|SE < \mu < \mu + |z^*|SE \quad (291)$$

$|z^*| = 1.96$ for a confidence level of 95%.

The origin of Eq.(291) is explained in the next section entitled “Confidence Intervals”. Confidence intervals are derived from the Gaussian in Fig.13, which should not be confused with the Gaussian of Fig.12. They are different!

0.38 Confidence Intervals

Normal distribution with mean μ and standard deviation σ :

$$\mathcal{N}(x; \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}. \quad (292)$$

Standard Normal Distribution (SND):

$$P(z) = \mathcal{N}(z; 0, 1) \quad (293)$$

Cumulative distribution for $P(z)$:

$$\Phi(z) = \int_{-\infty}^z dz' P(z') . \quad (294)$$

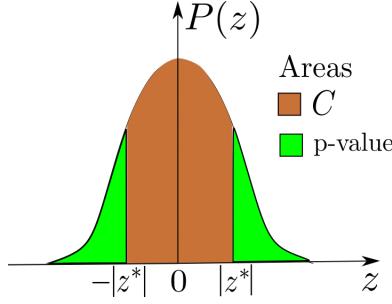


Figure 13: Interpretation of confidence level C and p-value as areas under curve of the Standard Normal Distribution (SND).

Confidence Level C and corresponding $|z^*|$ value (see Fig.13):

$$C = \int_{-|z^*|}^{|z^*|} dz P(z) = \Phi(|z^*|) - \Phi(-|z^*|) = 2 \left(\Phi(|z^*|) - \frac{1}{2} \right) \quad (295)$$

Equivalent definition:

$$C = P \left(\underbrace{\frac{|\underline{x} - \mu|}{\frac{\sigma}{\sqrt{n}}}}_{|z|} < |z^*| \right) \quad (296)$$

For $C = 95\%$, $|z^*| = 1.960 \approx 2$. For $C = 99\%$, $|z^*| = 2.576$.

Area of each tail in Fig.13 is usually called α , and the area of both tails is called the **p-value**:

$$C + \underbrace{2\alpha}_{p\text{-value}} = 1. \quad (297)$$

Estimators¹⁴ of mean μ and standard deviation σ from measurements x^σ of a sub-population Σ_1 of size $n = |\Sigma_1|$:

$$\hat{\mu} = \bar{x} = \frac{1}{n} \sum_{\sigma \in \Sigma_1} x^\sigma \quad (298)$$

$$\hat{\sigma}^2 = \frac{1}{n-1} \sum_{\sigma \in \Sigma_1} (x^\sigma - \bar{x})^2 \quad (299)$$

We get from Eq.(296), the **Error bars (a.k.a. confidence intervals)** and **Error E (aka margin of error)**:

¹⁴Don't confuse the sample index σ with the standard deviation σ .

$$\text{estimate of } x \text{ with error bars} = \bar{x} \pm |z^*| \underbrace{\frac{\hat{\sigma}}{\sqrt{n}}}_{E} \quad (300)$$

$$n = \left(\frac{|z^*| \hat{\sigma}}{E} \right)^2 \quad (301)$$

So far, we have assumed that the sub-population (aka sample population) is normally distributed. This might be false for several reasons. Some red flags: (1) n is too small (according to a rule of thumb derived from Central Limit Theorem, n should be larger than 30 to insure a Normal Distribution). (2) Sub-population not truly random (i.i.d.) because was taken without replacement. In many cases, especially when $n < 30$, the Student's t-distribution models the sub-population statistics much better than the Normal distribution.

The Student's t-distribution $\text{Stud}(t; \nu = n - 1)$, depends on a parameter ν called the number of degrees of freedom. In the case being considered here, ν equals the sub-population size n minus one. When fitting the data with $\text{Stud}()$, variable t replaces variable z , and $\text{Stud}(t; \nu = n - 1)$ replaces the Standard Normal distribution (SND) $\mathcal{N}(z; \mu = 0, \sigma = 1)$. $\text{Stud}()$ is symmetric about the origin like SND, but its tails are fatter. When fitting the data with $\text{Stud}()$, the $|z^*|$ value is replaced by a $|t^*|$ value. Eq.(295) is replaced by

$$C = \int_{-|t^*|}^{|t^*|} dt \text{Stud}(t) = \Phi_S(|t^*|) - \Phi_S(-|t^*|) = 2 \left(\Phi_S(|t^*|) - \frac{1}{2} \right), \quad (302)$$

where $\Phi_S()$ is the cumulative distribution for $\text{Stud}()$. Also, Eq.(300) is replaced by

$$\text{estimate of } x \text{ with error bars} = \bar{x} \pm |t^*| \underbrace{\frac{\hat{\sigma}}{\sqrt{n}}}_{E}. \quad (303)$$

Tables of $|t^*|(C, \nu = n - 1)$ are available. Note that $|t^*|$ depends on both C and ν , whereas $|z^*|(C)$ depends only on C .

0.39 p-value, general definition

Given a parameter θ , call $\theta = \theta_0$ (or $\theta < \theta_0$ or $\theta > \theta_0$) the **null hypothesis** h_0 , and call the negation of h_0 the **alternative hypothesis** h_1 . Assume we are given data $\vec{x} = \{x^\sigma | \sigma \in \Sigma\}$. Assume also that we are given distributions $P(\underline{x} = x|h)$ for $h \in \{h_0, h_1\}$, and $P(\underline{x} = x)$. Now let

$$P(\vec{x}|h) = \prod_{\sigma} P(\underline{x} = x^\sigma|h) \quad (304)$$

$$P(\vec{x}) = \prod_{\sigma} P(\underline{x} = x^{\sigma}) \quad (305)$$

(so the x^{σ} are i.i.d.).

A Bayesian would assume that there is a prior $P(h)$, and use it to calculate $P(h|\vec{x}) = \frac{P(\vec{x}|h)P(h)}{P(\vec{x})}$. $P(\underline{h} \neq h_0|\vec{x})$ is the probability that the null hypothesis is false. A p-value is a monotonically increasing function of $P(\underline{h} \neq h_0|\vec{x})$, so Bayesians have no trouble saying that **a p-value is a measure of $P(\underline{h} \neq h_0|\vec{x})$** , i.e., **a measure of the probability that the null-hypothesis is false**.

Frequentists, on the other hand, believe that h is a “parameter”, not a random variable, so $P(\underline{h} \neq h_0|\vec{x})$ is undefined. Next, we explain the correct way of thinking about p-values, according to Frequentists. p-values were invented by Frequentists, so it's worth hearing what they have to say about them. The Frequentist definition is not against Bayesianism, and Bayesians, unlike Frequentists, don't accuse Frequentists of having a sinfully incorrect definition of p-values. A Bayesian would just say: our definition of p-values (shown in red above) is not incorrect, but the Frequentist definition is more precise than ours, and doesn't assume a particular form for a prior. We welcome it.

Call the random variable \underline{t} the **test statistic** and the function $OTS(\vec{x})$ the **observed test statistic**. $OTS(\vec{x})$ is designed so that it becomes 0 when h_0 is exactly satisfied, and its magnitude grows monotonically as h_0 becomes less well satisfied. Frequentists define the **p-value** p as

$$p = \begin{cases} P(\underline{t} \geq OTS(\vec{x})|h_0) & \text{right-sided-tail, if } h_0 \text{ is } \theta < \theta_0 \\ P(\underline{t} \leq OTS(\vec{x})|h_0) & \text{left-sided-tail, if } h_0 \text{ is } \theta > \theta_0 \\ P(|\underline{t}| > |OTS(\vec{x})| |h_0) & \text{double-sided-tail, if } h_0 \text{ is } \theta = \theta_0 \end{cases} \quad (306)$$

Thus, for a Frequentist, **a p-value is a probabilistic measure of the size of the region where the h_0 hypothesis is extremely violated**. If that region is large, then the region where the h_0 hypothesis is approximately satisfied is small, which means the h_0 hypothesis is approximately satisfied. The smaller the p-value, the less likely it is that the h_0 hypothesis is true, just like the Bayesian definition says. Note that the p-value is a probability so it ranges in value from 0 to 1.

Suppose we are given a subpopulation with n samples, mean \bar{x} and variance $\hat{\sigma}$. Let $\theta_0 = \mu_0$. For the z-test and the t-test, we define

$$\underline{t} = \underline{z} = \frac{\underline{x} - \mu}{\frac{\hat{\sigma}}{\sqrt{n}}}, \quad OTS(\vec{x}) = z^*, \quad (307)$$

where the **z-score** z^* is defined by

$$z^* = \frac{\bar{x} - \mu_0}{\frac{\hat{\sigma}}{\sqrt{n}}}. \quad (308)$$

For a **z-test**,

$$P(\underline{z} \geq z^* | h_0) = \Phi(z^*) \quad \text{if } h_0 \text{ is } \mu < \mu_0 \quad (309a)$$

$$P(\underline{z} \leq z^* | h_0) = 1 - \Phi(z^*) = \Phi(-z^*) \quad \text{if } h_0 \text{ is } \mu > \mu_0 \quad (309b)$$

$$P(|\underline{z}| \geq |z^*| | h_0) = 2\Phi(-|z^*|) \quad \text{if } h_0 \text{ is } \mu = \mu_0 \quad (309c)$$

where $\Phi(x)$ is the cumulative distribution for the Standard Normal Distribution $\mathcal{N}(x; \mu = 0, \sigma = 1)$. For a **t-test**, $\Phi()$ is replaced by $\Phi_S()$, where $\Phi_S()$ is the cumulative distribution for the Student t-distribution $\text{Stud}(x; \nu = n - 1)$. Note that Eq.(309c) agrees with Eq.(296).

0.40 Short Summary of Boolean Algebra

See Ref.[75] for more info about this topic.

Suppose $x, y, z \in \{0, 1\}$. Define

$$x \text{ or } y = x \vee y = x + y - xy , \quad (310)$$

$$x \text{ and } y = x \wedge y = xy , \quad (311)$$

and

$$\text{not } x = \bar{x} = 1 - x , \quad (312)$$

where we are using normal addition and multiplication on the right hand sides.¹⁵

Actually, since $x \wedge y = xy$, we can omit writing the symbol \wedge . The symbol \wedge is useful to exhibit the symmetry of the identities, and to remark about the analogous identities for sets, where \wedge becomes intersection \cap and \vee becomes union \cup . However, for practical calculations, \wedge is an unnecessary nuisance.

Since $x \in \{0, 1\}$,

$$P(\bar{x}) = 1 - P(x) . \quad (313)$$

Clearly, from analyzing the simple event space $(x, y) \in \{0, 1\}^2$,

$$P(x \vee y) = P(x) + P(y) - P(x \wedge y) . \quad (314)$$

¹⁵Note the difference between \vee and modulus 2 addition \oplus . For \oplus (aka XOR): $x \oplus y = x + y - 2xy$.

Associativity	$x \vee (y \vee z) = (x \vee y) \vee z$ $x \wedge (y \wedge z) = (x \wedge y) \wedge z$
Commutativity	$x \vee y = y \vee x$ $x \wedge y = y \wedge x$
Distributivity	$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$ $x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$
Identity	$x \vee 0 = x$ $x \wedge 1 = x$
Annihilator	$x \wedge 0 = 0$ $x \vee 1 = 1$
Idempotence	$x \vee x = x$ $x \wedge x = x$
Absorption	$x \wedge (x \vee y) = x$ $x \vee (x \wedge y) = x$
Complementation	$x \wedge \bar{x} = 0$ $x \vee \bar{x} = 1$
Double negation	$\overline{(\bar{x})} = x$
De Morgan Laws	$\bar{x} \wedge \bar{y} = \overline{(x \vee y)}$ $\bar{x} \vee \bar{y} = \overline{(x \wedge y)}$

Table 1: Boolean Algebra Identities

Definition of a Bayesian Network

A **directed graph** $G = (V, E)$ consists of two sets, V and E . V contains the **vertices (nodes)** and E contains the **edges (arrows)**. An arrow $\underline{a} \rightarrow \underline{b}$ is an ordered pair $(\underline{a}, \underline{b})$ where $\underline{a}, \underline{b} \in V$.

The **parents** of a node \underline{x} are those nodes \underline{a} such that there are arrows $\underline{a} \rightarrow \underline{x}$. The **children** of a node \underline{x} are those nodes \underline{b} such that there are arrows $\underline{x} \rightarrow \underline{b}$. A **root node** is a node with no parents. A **leaf node** is a node with no children. The **neighbors** of a node \underline{x} is the set of parents and children of \underline{x} .

A **path** is a set of nodes that are connected by arrows, so that all nodes have 1 or 2 neighbors, but only two nodes (**open path**) or zero nodes (**closed path**) have only one neighbor. A **directed path** is a path in which all the arrows point in the same direction. A **loop** is a closed path; i.e., a path in which all nodes have exactly 2 neighbors. A **cycle** is a directed loop. A **Directed Acyclic Graph (DAG)** is a directed graph that has no cycles.

A **fully connected directed graph** is a directed graph in which every node has all other nodes as neighbors. Figs.14 and 15 show 2 different ways of drawing the same directed graph, a fully connected graph with 4 nodes. Note that a convenient way to label the nodes of a fully connected directed graph with N nodes is to point arrows from \underline{x}_k to \underline{x}_j where $j = 0, 1, 2, \dots, N - 1$ and $k = j - 1, j - 2, \dots, 0$.



Figure 14: Fully connected directed graph with 4 nodes, drawn as a line.

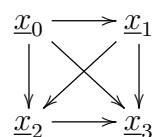


Figure 15: Fully connected directed graph with 4 nodes, drawn as a square.

A **connected graph** is a graph for which there is no way of separating the nodes into two sets so that there is no arrow from one set to the other. A **tree** is a

directed graph in which all nodes have a single parent except for a single node called the “root” node which has no parents. A **polytree** is a DAG with no loops.

A **Bayesian network (bnet)** consists of a DAG and a **Transition Probability Matrix (TPM)** associated with each node of the graph. A TPM is often called a **Conditional Probability Table (CPT)**. The **structure** of a bnet is its DAG alone, sans the TPMs. The **skeleton** of a bnet is the undirected graph beneath the bnet’s DAG.

In this book, random variables are indicated by underlined letters and their values by non-underlined letters. We use $S_{\underline{x}}$ to denote the set of states (i.e., values) that a random variable \underline{x} can assume. Each node of a bnet is labelled by a random variable. Thus, $\underline{x} = x$ means that node \underline{x} is in state x .

Some sets of nodes associated with each node \underline{a} of a bnet

- $ch(\underline{a})$ = children of \underline{a} .
- $pa(\underline{a})$ = parents of \underline{a} .
- $nb(\underline{a}) = pa(\underline{a}) \cup ch(\underline{a})$ = neighbors of \underline{a} .
- $de(\underline{a}) = \bigcup_{n=1}^{\infty} ch^n(\underline{a}) = ch(\underline{a}) \cup ch \circ ch(\underline{a}) \cup \dots$, descendants of \underline{a} .
- $an(\underline{a}) = \bigcup_{n=1}^{\infty} pa^n(\underline{a}) = pa(\underline{a}) \cup pa \circ pa(\underline{a}) \cup \dots$, ancestors of \underline{a} .

In this book, we will use $\underline{a}.$ to indicate a **multi-node (node set, node array)** $\underline{a}. = (\underline{a}_j)_{j=0,1,\dots,na-1}$. We will often treat multinodes as if they were sets, and combine them with the usual set operators. For instance, for two multinodes $\underline{a}.$ and $\underline{b}.$, we define $\underline{a}. \cup \underline{b}.$, $\underline{a}. \cap \underline{b}.$, $\underline{a}. - \underline{b}.$ and $\underline{a}. \subset \underline{b}.$ in the obvious way. We will indicate a singleton set (single node multi-node) $\underline{a}. = \{\underline{a}\}$ simply by $\underline{a}. = \underline{a}.$ For instance, $\underline{a}. - \underline{b} = \underline{a}. - \{\underline{b}\}.$

The TPM of a node \underline{x} of a bnet is a matrix of probabilities $P(\underline{x} = x | pa(\underline{x}) = a.)$, where $x \in S_x$ and $a. \in S_{pa(\underline{x})}$.

A **deterministic node** is a node such that its TPM is of the form

$$P(\underline{x} = x | pa(\underline{x}) = a.) = \delta(x, f(a.)) \quad (315)$$

for some function $f : S_{pa(\underline{x})} \rightarrow S_x$, where $\delta(x, y)$ is the Kronecker delta function.

A bnet with nodes $\underline{x}.$ represents a probability distribution

$$P(x.) = \prod_j P(\underline{x}_j = x_j | (\underline{x}_k = x_k)_{k: \underline{x}_k \in pa(\underline{x}_j)}). \quad (316)$$

Note that for a fully connected bnet with N nodes, Eq.(316) becomes

$$P(x.) = \prod_{j=0}^{N-1} P(x_j | (x_k)_{k=j-1,j-2,\dots,0}). \quad (317)$$

For example, if $N = 4$, Eq.(317) becomes

$$P(x_0, x_1, x_2, x_3) = P(x_3|x_2, x_1, x_0)P(x_2|x_1, x_0)P(x_1|x_0)P(x_0) . \quad (318)$$

We see that Eq.(317) is just the chain rule for conditional probabilities.

In this book, we often use the following conventions for bnet instantiations:

Random variables are underlined and their values are not. For example, $\underline{a} = a$ means the random variable \underline{a} takes the value a . Diagrams with nodes that are underlined represent Bayesian Networks (bnets) and the same diagram with the letters not underlined represents a specific **instantiation** of that bnet. For example $\underline{a} \rightarrow \underline{b}$ represents the bnet with conditional probability distribution $P(b|a)$, whereas $a \rightarrow b$ represents $P(b|a)$ itself.

If \underline{a} is a root node, then $\sum \underline{a}$ signifies a weighted sum $\sum_a P(a)$. For example, $\sum \underline{a} \rightarrow b = \sum_a P(a)P(b|a)$. If a is not a root node as in $x \rightarrow \sum \underline{a} \rightarrow y = \sum_a P(y|a)P(a|x)$, then $\sum \underline{a}$ signifies a simple unweighted sum \sum_a .

In this book, Unobserved nodes are indicated by enclosing them in a dashed circle. For example, (\underline{u}) .

Given an arbitrarily large dataset of samples for the random variables $(x_i)_{i=0,1,\dots,N-1}$, there may be several bnets (differing in the direction of some arrows) that fit the data well. However, according to Pearl's causality theory, only one of these bnets is used by Nature. I like to refer to that single one as the **causally correct (CC) Bayesian network**.¹⁶ ¹⁷

It's important to realize that bnets that are not CC are far from useless; they are frequently used as intermediate calculational tools. They are incorrect causally, but they aren't incorrect numerically. For instance, in Bayes rule, we switch from the CC bnet $P(x|\theta) = x \leftarrow \theta$ to the non-CC bnet $P(\theta|x) = x \rightarrow \theta$, where x is the data and θ are the parameters.

¹⁶The uniqueness of a CC bnet can be taken to be an implicit axiom of causality theory. Alternatively, instead of assuming uniqueness, one can assume that we are using a DAG which is, according to some measure of goodness of causal fit, the best one among the DAGs of a given set of DAGs. (see Chapter 28).

¹⁷We won't use the term "causal bnet" in this book. To avoid confusion, we will use the term "causally correct (CC) bnet" instead. Pearl sometimes defines a causal bnet to be a CC bnet that is also a "SCM" (i.e., a bnet whose internal nodes are deterministic and external ones are probabilistic.)

Bayesian Networks, Causality and the Passage of Time

This chapter is based on a blog post (see Ref.[57]) from my blog “Quantum Bayesian Networks”.

0.41 Unifying Principle of this book

The unifying principle of this book is Bayesian Networks (bnets). The main goal of this book is to explain as much of Artificial Intelligence (AI) and Machine Learning (ML) as possible using bnets.

Bayesian Networks are a graphical representation of the chain rule for conditional probabilities. They are not a “heuristic algorithm” like XGBoost or Neural Nets. They are a very simple, intuitive, basic and general definition. I would say that the definition of a Bayesian Network is as important to Probability Theory as the definition of a Group is to Abstract Algebra. Algebraic groups are never going to go out of fashion and neither are B nets.

An Artificial Neural Net can be defined as a Bayesian Network with a layered structure, and such that all its nodes are deterministic¹⁸. A decision tree is not exactly a Bayesian Network, but it can be trivially replaced by an equivalent B net that has the same tree structure (for more details about this equivalence, see the Chapter 13 on decision trees). The SCM diagrams favored by Pearl are just bnets whose internal nodes are deterministic and external ones are probabilistic. In fact, as I show in this book, most methods in AI can be understood in terms of B nets—just like many theorems in Abstract Algebra can be understood in terms of groups.

0.42 You say tomato, I say tomato

In this chapter, I will use the terms Bayesian Network (bnet), causal model and DAG as if they were synonymous. My justification for doing this is as follows.

¹⁸NNs are DAGs, but they contain a lot of extra, spurious nodes with no causal motivation. So I like to say that NNs are acausal DAGs.

A Bayesian Network is a DAG + probability tables. One can easily compute the probability tables from DAG + Dataset. Therefore,

You say DAG+Dataset, I say Bayesian Network.

The use of the terms “causal model” and “DAG”, as an alternative to the term “Bayesian Network”, has become popular in the last decade among economists, epidemiologists, AI researchers and even Judea Pearl himself. It seems some people think “causal models” and “DAGs” are revolutionary, whereas Bayesian Networks are a concept that was tried 25 years ago, and has been replaced since then by stuff that works better. But any time you have a Dataset, which is almost always true in practice in Economics, Epidemiology and AI, a DAG implies a Bayesian Network and vice versa.

0.43 A dataset is causal model free

Time and time again, Judea Pearl makes the point on Twitter to neural net advocates that they are trying to do a provably impossible task, to derive a causal model from data. I could be wrong, but this is what I think he means.

When Pearl says “data”, he is referring to what is commonly called a dataset. A dataset is a table of data, where all the entries of each column have the same units, and measure a single feature, and each row refers to one particular sample or individual. Datasets are particularly useful for estimating probability distributions and for training neural nets. When Pearl says a “causal model”, he is referring to a DAG (directed acyclic graph) or a bnet (Bayesian Network= DAG + probability table for each node of DAG).

Sure, you can try to derive a causal model from a dataset, but you’ll soon find out that you can only go so far.

The process of finding a partial causal model from a dataset is called structure learning (SL). SL can be done quite nicely with Marco Scutari’s open source program bnlearn. There are 2 main types of SL algorithms: score-based and constraint based. The first and still very competitive constraint-based SL algorithm was the Inductive Causation (IC) algorithm proposed by Pearl and Verma in 1991. So Pearl is quite aware of SL. The problem is that SL often cannot narrow down the causal model to a single one. It finds an undirected graph (UG), and it can determine the direction of some of the arrows in the UG, but it is often incapable, for well understood fundamental—not just technical—reasons, of finding the direction of ALL the arrows of the UG. So it often fails to fully specify a DAG model.

Let’s call the ordered pair (dataset, causal model) a dataset++ . Then what I believe Pearl is saying is that a dataset is causal model-free or causal model-less (although sometimes one can find a partial causal model hidden in there). A dataset is not a dataset++.

0.44 What is causality?

What is Causality, really, and how do Bayesian Networks (aka Causal Models, DAGs) encode it?

For me, Causality is a time-induced ordering between two events, the transmission of information (and its accompanying energy) from the earlier of the two events to the later one, and the physical response of the later event to the reception of that information. The nodes of a bnet represent random variables. Some of those random variables are clearly events (i.e., they occur at a definite time). For example, let $D=0$ if a patient is not given a drug, $D=1$ if he/she is given it. D occurs at a definite time. But other random variables represent qualities which do not occur at a definite time. For example, $G=\text{gender}=\text{male}, \text{female}$. G does not occur at a definite time. But even in the case of a quality like G , its value is first decided at birth, so one can ascribe to G a particular, albeit fuzzy time interval during which it is decided. If $M=0(\text{single}), 1(\text{married})$, then we can assign to M the day of the marriage. Both the time interval assigned to G and to M are somewhat ambiguous, but still, most people would say that G occurs before M (if a marriage occurs at all). Saying the opposite, that M occurs before G , seems pretty hard to understand. If two nodes A and B of a bnet have time intervals ascribed to them such that the time interval of A does not clearly occur before or after the time interval of B , and if also there is a large correlation between A and B , then it probably does not matter much whether one draws an arrow from A to B , or the opposite.

Now that we understand that the arrows in a bnet really do encode the direction of time, it becomes clear why a dataset does not fully specify a bnet. By a dataset (think of a dataframe in Pandas or R), I mean an array of numbers where the columns refer to features and the rows refer to individuals in a population. The column labels of the dataset become the node names of the bnet. Nowhere in a dataset is there any indication of the time ordering of the features. Hence, it's impossible to create, from a dataset alone, a bnet, because bnets do carry such time-ordering information.

Chapter 29 discusses Granger Causality (GC). The critics of GC point out that it assumes, somewhat erroneously, that if event A precedes B and the two events are correlated, A must cause B. I agree. Most roosters crow in response to the stimulus of the sunrise light. A rooster could crow before sunrise if, for example, he had an alarm clock that woke him up 30 minutes before sunrise, but such cases are uncommon, and seem to involve other intermediate events. The moral is that time ordering and correlation are necessary but not sufficient conditions for causality. To establish causality with more certainty, one also needs a pinch of prior expert knowledge, or one must gain that expert knowledge through “do” operator experimentation.

0.45 Bayesian Networks and the passage of time

Now that we understand that a bnet's arrows are encoding roughly the passage of time, it becomes possible to glean from this insight a simple method, which, although not very rigorous, is really helpful to me. I will illustrate said method with the famous “Asia” bnet in Fig.16. In this bnet, all nodes have two possible values, 0 and 1.

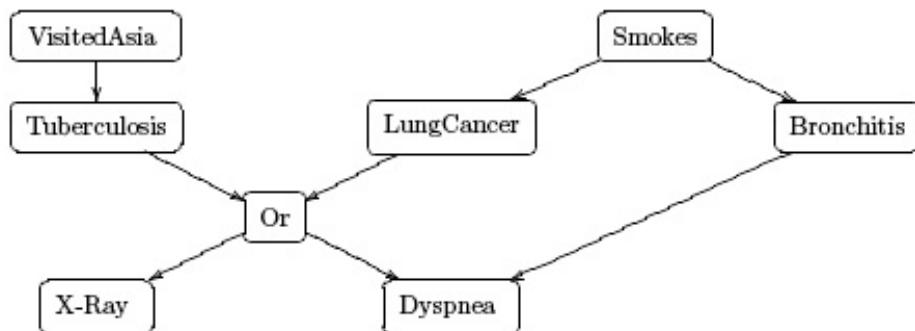


Figure 16: Asia bnet. Dyspnea=trouble breathing

Given a dataset for this bnet, one can calculate the correlation between every 2 features of the dataset. The feature names become the node names, and links are drawn between any 2 nodes whose correlation is greater in absolute value to some threshold value. This gives an undirected graph that can be obtained from bnet Fig.16 by erasing the directions of the arrows. So how can we guess the directions of the arrows? Well, one uses a little bit of “expert knowledge” to conclude that

$$\text{time(Visited Asia)} < \text{time(Tuberculosis)} < \text{time(Or)} < \text{time(X-Ray, Dyspnea)}$$

Also

$$\text{time(smokes)} < \text{time(LungCancer, Bronchitis)} < \text{time(Or)} < \text{time(Dyspnea)}$$

If $\text{time}(A) < \text{time}(B)$, then $A \rightarrow B$. Like I said before, the times we ascribe to these events are somewhat fuzzy and open to debate, so this algorithm is far from being rigorous. But often, saying that $\text{time}(A) < \text{time}(B)$ makes much more sense than saying that $\text{time}(B) < \text{time}(A)$. When in doubt about the best direction to give to an arrow of an undirected graph, I recommend calculating a Goodness of Causal Fit metric (see Chapter 28) which makes use of “do” operator experimentation.

A dataset cannot fully specify a bnet because it lacks time ordering info. A dataset also cannot do the harder task of specifying a bnet that is a good causal fit to the problem, because it lacks time ordering info AND prior expert knowledge AND expert knowledge gained from posterior “do” operator experimentation.

0.46 Advice for the DAG-phobic

DAGs are your friends. DAGs should be easy and fun to dream up. After all, I am convinced that DAGs are an integral part of how humans think, so they should come naturally to us. Nevertheless, many people are scared of, or detest, DAGs. I think it's because they fail to grasp the following 3 things:

1. DAGs are not unique. Stop thinking that you have to find the unique DAG for the situation being considered. You just have to find a DAG that is a good causal fit for the situation. If a DAG is too complicated, you can always simplify it by merging several nodes into a single more abstract one, or by summing over unwanted nodes.
2. DAGs are roughly ordered from past to present. The arrows of a DAG roughly reflect the passage of time.
3. DAGs represent a scientific hypothesis that can and should be tested with do experiments.

Chapter 1

AdaBoost

This chapter is based on Ref.[65].

Adaptive Boosting (AdaBoost) is a method of constructing a strong classifier function as a linear combination of an ensemble of weak classifier functions.

Below, we will abbreviate “ensemble classifiers” by “e-classifiers” and “weak classifiers” by “w-classifiers”.

Chapter 13 defines decision trees (dtrees) and explains how to construct them. A **tree stump** is a dtree with only one parent and 2 children nodes. Usually the w-classifier functions for AdaBoost come from tree stumps (because tree stumps are w-classifiers and simple to compute), but the core AdaBoost algorithm is oblivious to where the w-classifier functions came from.

Boosting (see this chapter on AdaBoost and Chapter 80 on XGBoost) and bagging (see Chapter 58 on Random Forest) are two methods of building a classifier function from an ensemble of classifier functions. These two methods are most commonly applied to dtrees: Boosting for an ensemble of small dtrees, and Bagging for a random forest (which is an ensemble of dtrees that are usually much more complicated than small dtrees).

1.1 AdaBoost for general ensemble of w-classifiers

In this section we discuss the core AdaBoost algorithm, valid for a generic ensemble of w-classifiers.

Let $L = [0, 1, 2, \dots, nsam - 1]$ be a list of individuals (samples) in a population. In this chapter, we will use the notation $A^\sigma = A[\sigma]$ and $\vec{A} = [A^\sigma : \sigma \in L]$ for a list (vector, 1-D array) indexed by L . We will refer to $DS = (\vec{x}, \vec{y})$ where $x^\sigma \in S_{\underline{x}}$, $y^\sigma \in S_y$, as a dataset. Let $T = \{0, 1, \dots, nt - 1\}$. Let $x^\sigma = (x_0^\sigma, x_1^\sigma, \dots, x_{nt-1}^\sigma) \in S_{\underline{x}_0} \times S_{\underline{x}_1} \times \dots \times S_{\underline{x}_{nt-1}} = S_{\underline{x}}$.

AdaBoost assumes that the classifier class set S_y and all the feature sets $S_{\underline{x}_t}$ are binary: $S_y = S_{\underline{x}_t} = \{-1, 1\}$ for all $t \in T$.

Suppose that we are given an ensemble of nt **w-classifiers** $Y_t : S_{\underline{x}} \rightarrow \{-1, 1\}$, where $t \in T$. Suppose we want to find **intermediate e-classifiers** $\mathcal{Y}_t : S_{\underline{x}} \rightarrow \mathbb{R}$ given

by

$$\mathcal{Y}_t(x^\sigma) = \sum_{t'=0}^t \alpha_{t'} Y_{t'}(x^\sigma) \in \mathbb{R} \quad (1.1)$$

for $t \in T$ and a **final e-classifier** given by

$$\mathcal{Y}_{fin}(x^\sigma) = \text{sign}(\mathcal{Y}_{nt-1}(x^\sigma)) \in \{-1, 1\}. \quad (1.2)$$

The AdaBoost algorithm yields a set of coefficients α_t for which the final e-classifier is much stronger (i.e., less error prone) than any of the w-classifiers of the ensemble.

Note that

$$\mathcal{Y}_t(x^\sigma) = \mathcal{Y}_{t-1}(x^\sigma) + \alpha_t Y_t(x^\sigma) \quad (1.3)$$

for $t \in T$ if we define $\mathcal{Y}_{-1} = 0$. Hence

$$\underbrace{e^{-y^\sigma \mathcal{Y}_t(x^\sigma)}}_{Z_t w_{t+1}^\sigma} = \underbrace{e^{-y^\sigma \mathcal{Y}_{t-1}(x^\sigma)}}_{Z_{t-1} w_t^\sigma} e^{-\alpha_t y^\sigma Y_t(x^\sigma)} \quad (1.4)$$

where the **weights** w_t^σ and the **partition function** Z_t are defined by

$$w_{t+1}^\sigma = \begin{cases} 1/nsam & \text{for } t = -1 \\ \frac{\exp(-y^\sigma \mathcal{Y}_t(x^\sigma))}{Z_t} & \text{for } t \geq 0 \end{cases} \quad (1.5)$$

and

$$Z_t = \sum_\sigma e^{-y^\sigma \mathcal{Y}_t(x^\sigma)}. \quad (1.6)$$

Note that the probability distribution $P(\sigma|t+1) = w_{t+1}^\sigma$ of weights at time $t+1$ emphasizes (i.e., gives higher probability to) the errors (i.e., occurrences of $y^\sigma \mathcal{Y}_t(x^\sigma) = -1$ for some population individual σ) of the previous (i.e., at time t) intermediate e-classifier \mathcal{Y}_t . In other words, every new intermediate e-classifier \mathcal{Y}_{t+1} concentrates on those individuals σ on which the previous e-classifier \mathcal{Y}_t performed poorly.

Note also that the partition function Z_t is a good measure of the **classification error** (i.e., occurrences of $y^\sigma \mathcal{Y}_t(x^\sigma) = -1$) of \mathcal{Y}_t . We will therefore use Z_t for that purpose, as an error measure.

For $t > 1$, we have

$$Z_t = \sum_\sigma e^{-y^\sigma \mathcal{Y}_t(x^\sigma)} \quad (1.7)$$

$$= \sum_\sigma \underbrace{e^{-y^\sigma \mathcal{Y}_{t-1}(x^\sigma)}}_{Z_{t-1} w_t^\sigma} e^{-\alpha_t y^\sigma Y_t(x^\sigma)} \quad (1.8)$$

$$= Z_{t-1} E_\sigma [e^{-\alpha_t y^\sigma Y_t(x^\sigma)}] \quad (1.9)$$

Define the **success rate** by

$$S_t = \sum_{\sigma} w_t^{\sigma} \mathbb{1}(y^{\sigma} Y_t(x^{\sigma}) = 1) \quad (1.10)$$

$$= E_{\sigma}[\underbrace{\mathbb{1}(y^{\sigma} Y_t(x^{\sigma}) = 1)}_{\text{iff } y^{\sigma} = Y_t(x^{\sigma})}] \quad (1.11)$$

and the **failure rate** by

$$F_t = \sum_{\sigma} w_t^{\sigma} \mathbb{1}(y^{\sigma} Y_t(x^{\sigma}) = -1) \quad (1.12)$$

$$= E_{\sigma}[\underbrace{\mathbb{1}(y^{\sigma} Y_t(x^{\sigma}) = -1)}_{\text{iff } y^{\sigma} \neq Y_t(x^{\sigma})}] . \quad (1.13)$$

Note that

$$S_t + F_t = 1 , \quad (1.14)$$

and

$$Z_t = Z_{t-1}(e^{-\alpha_t} S_t + e^{+\alpha_t} F_t) . \quad (1.15)$$

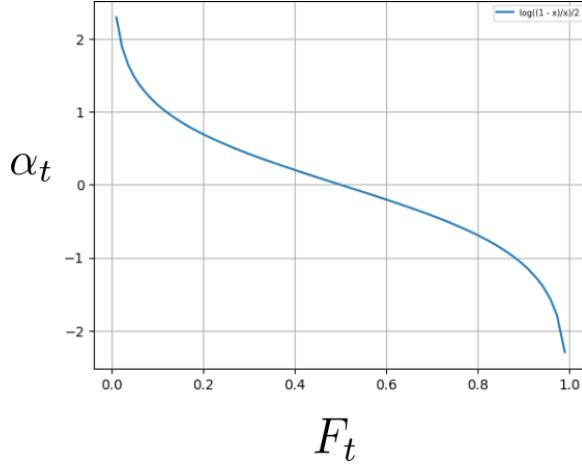


Figure 1.1: Plot of function $\alpha_t = \frac{1}{2} \ln \frac{1-F_t}{F_t}$.

We can find the α_t values that minimize the classification error Z_t , and then evaluate Z_t at those optimum α_t values:

$$\frac{dZ_t}{d\alpha_t} = Z_{t-1}(-e^{-\alpha_t} S_t + e^{\alpha_t} F_t) = 0 \quad (1.16)$$

$$e^{2\alpha_t} = \frac{S_t}{F_t} \quad (1.17)$$

$$\alpha_t = \frac{1}{2} \ln \frac{S_t}{F_t} = \frac{1}{2} \ln \frac{1 - F_t}{F_t} \quad (1.18)$$

$$\frac{Z_t}{Z_{t-1}} = 2\sqrt{S_t F_t} = 2\sqrt{(1 - F_t) F_t} \leq 1 \quad (1.19)$$

$f(x) = 2\sqrt{(1 - x)x}$ for $x \in [0, 1]$ is dome shaped and its maximum is 1, which is achieved iff $x = \frac{1}{2}$

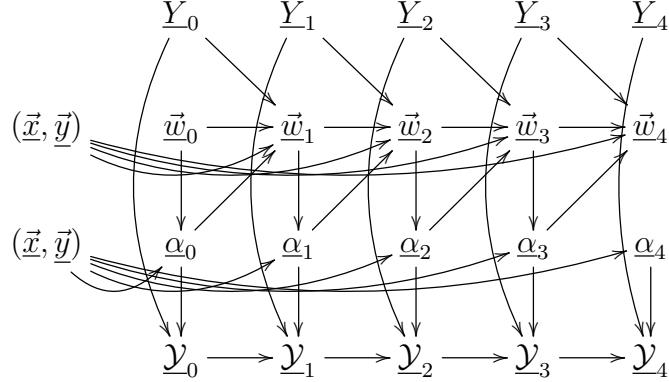


Figure 1.2: Bnet for AdaBoost with 5 w-classifiers, $nt = 5$. All nodes labelled (\vec{x}, \vec{y}) are the same node.

The AdaBoost algo described in this chapter is summarized by the bnet of Fig.1.2. The TPMs, printed in blue, for that bnet, are as follows:

$$P(w_0^\sigma) = \frac{1}{nsam} \quad (1.20)$$

for all σ .

$$P(\vec{w}_{t+1} | \vec{w}_t, \alpha_t, Y_t, \vec{x}, \vec{y}) = \prod_\sigma \mathbb{1}(\quad w_{t+1}^\sigma = \frac{w_t^\sigma e^{-\alpha_t y^\sigma Y_t(x^\sigma)}}{\sum_\sigma \text{numerator}} \quad) \quad (1.21)$$

for $t \geq 0$.

$$P(\alpha_t | \vec{w}_t, \vec{x}, \vec{y}) = \mathbb{1}(\quad \alpha_t = \frac{1}{2} \ln \frac{1 - F_t}{F_t} \text{ where } F_t = F_t(\vec{w}_t, \vec{x}, \vec{y}) \quad) \quad (1.22)$$

for $t \in T$.

$$P(\mathcal{Y}_t | \mathcal{Y}_{t-1}, \alpha_t, Y_t) = \mathbb{1}(\quad \mathcal{Y}_t = \mathcal{Y}_{t-1} + \alpha_t Y_t \quad) \quad (1.23)$$

for $t \in T$, where $\mathcal{Y}_{-1} = 0$.

1.2 AdaBoost for ensemble of tree stumps

Keep in mind that AdaBoost assumes $S_y = S_{x_t} = \{-1, 1\}$ for all $t \in T = \{0, 1, \dots, nt - 1\}$. In order to implement AdaBoost, we need to specify nt w-classifiers $Y_t : \{-1, 1\}^{nt} \rightarrow \{-1, 1\}$ for $t \in T$. One can either specify the nt w-classifiers a priori or build them on-the-fly.

- **w-classifiers specified a priori**

Define a classifier for each feature x_t where $t \in T$ by:

$$Y_t(x^\sigma) = x_t^\sigma \in \{-1, 1\} \quad (1.24)$$

Hence, for this classifier, $y^\sigma Y_t(x^\sigma) = y^\sigma x_t^\sigma$.

- **w-classifiers built on-the-fly**

Recall dataset $(\vec{x}, \vec{y}) = [(x^\sigma, y^\sigma) : \sigma \in L]$ is indexed by the list $L = [0, 1, \dots, nsam - 1]$. If L_j is a list (possibly with duplicate items) such that $set(L_j) \subset set(L)$, then define $DS_j = (\vec{x}, \vec{y})_{L_j} = ((x^\sigma)_{\sigma \in L_j}, (y^\sigma)_{\sigma \in L_j})$. We will refer to DS_j as the **restriction of (\vec{x}, \vec{y}) to L_j** .

The idea behind on-the-fly w-classifiers is to choose $Y_t(x^\sigma) = x_t^\sigma$, where x_t is the feature with the lowest Gini in the current dataset $(\vec{x}, \vec{y})_{L_t}$. To build $(\vec{x}, \vec{y})_{L_t}$, we select at random from $L = [0, 1, \dots, nsam - 1]$, a list L_t of the same length as L , using the probability distribution \vec{w}_{t-1} . By choosing L_t with probabilities \vec{w}_{t-1} , we emphasize individuals σ that are failing. Then we define $(\vec{x}, \vec{y})_{L_t}$ as the restriction of (\vec{x}, \vec{y}) to L_t .

Perhaps a causal diagram will make all these new steps clearer to the reader. The bnet Fig.1.3 is a modification of the bnet Fig.1.2 to include these new steps. The TPMs, printed in blue, for new or changed nodes, are as follows:

$$P(Y_t | (\vec{x}, \vec{y})_{L_t}) = \mathbb{1}(\text{ } Y_t(x^\sigma) = x_t^\sigma \text{ where } x_t \text{ is feature in } (\vec{x}, \vec{y})_{L_t} \text{ the with lowest Gini. }) \quad (1.25)$$

$$P(L_{t+1}^\sigma = \sigma' | \vec{w}_t) = w_t^{\sigma'} \quad (1.26)$$

$$P((\vec{x}, \vec{y})_{L_t} | L_t, (\vec{x}, \vec{y})) = \mathbb{1}(\text{ } (\vec{x}, \vec{y})_{L_t} = \text{ restriction of } (\vec{x}, \vec{y}) \text{ to } L_t \text{ }) \quad (1.27)$$

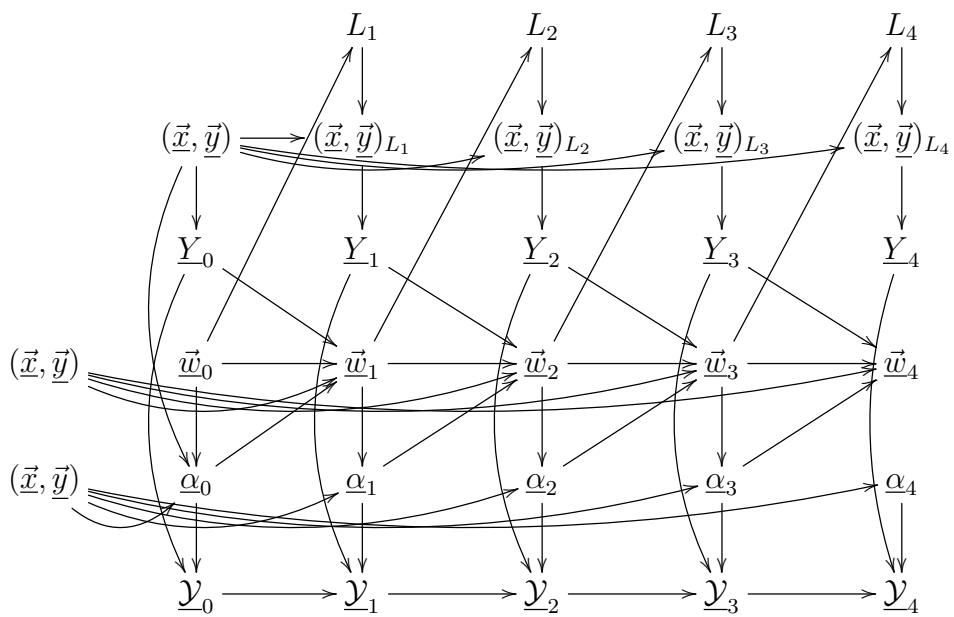


Figure 1.3: Modification of the bnet of Fig.1.2 to include on-the-fly generation of the w-classifiers.

Chapter 2

ARACNE structure learning

This chapter is based on Ref.[24].

The ARACNE algo is an algo for learning the structure of a bnet from data. The algo considers data samples for n random variables $(x_i)_{i=0,1,\dots,n-1}$, and estimates the mutual information $MI_{i,j} = H(\underline{x}_i : \underline{x}_j)$ between every pair of nodes. The set UG is initialized to contain all the edges of a fully connected undirected graph. Next the algo removes from UG every edge with $MI_{i,j} < \epsilon$ for some threshold $0 < \epsilon \ll 1$. Then the algo examines every triplet of edges in UG , and marks for removal the edge of the triplet with the smallest MI. Finally, the algo removes from UG all edges marked for removal. Each triplet is analyzed irrespective of whether its edges have been marked for removal when considering a prior triplet. Thus the network constructed by the algorithm is independent of the order in which the triplets are examined. Some of the unoriented edges in UG can be given an orientation using the same techniques used to orient edges in constraint based structure learning (see Chapter 70).

Ref.[24] incorrectly claims that removing the smaller of 3 MI's is "an application" of the Data Processing Inequality (DPI) of Shannon Information Theory. See Chapter 41 for more info about DPI. Note that DPI is only valid for a Markov chain, and not all triplets of random variables are in a Markov chain. Removing the smaller of 3 numbers does not require DPI.

Fig.2.1 gives an example of the application of the ARACNE algo.

See Chapter 9 on Chow-Liu trees (CLT). A CLT is just a maximum spanning tree where the weights are mutual informations $MI_{i,j}$ estimated from data.

Sometimes, the outcome of the ARACNE algo is a CLT. For example, Fig.2.1 (a) was considered in Chapter 9 on CLTs, and the CLT algo also gave Fig.2.1 (c) as the final structure.

According to Ref.[24], the ARACNE algo sometimes yields a polytree (i.e., a connected graph with no loops). It may even yield a structure with loops. Hence, it does not always yield a CLT.

By breaking all cliques (i.e., fully connected subgraphs) with 3 edges and 3 nodes, ARACNE breaks all cliques with 3 or more nodes. However, cliques are not uncommon in Nature, especially 3 node cliques. Cliques become less likely in Nature

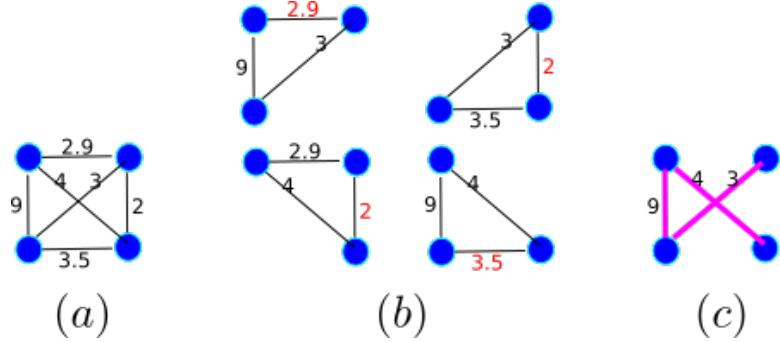


Figure 2.1: An example where the ARACNE algo gives a Chow-Liu tree. (a) Fully connected undirected graph with weights $MI_{i,j}$ along the edges. (b) All 4 possible triplets of edges with nonzero weights. Edges marked for removal have their weights printed in red.(c) Final structure.

the bigger the number of nodes they have *after* 3. Therefore, a nice generalization of ARACNE would be to list all 4 node cliques, and break each of them by eliminating their edge with the smallest MI. This will have the effect of breaking all cliques with 4 or more nodes but keeping 3 node cliques. One could also break some, not all, of the 3 node cliques, by consecutively removing the clique-breaking-edge with the smallest MI of all edges of all 3 node cliques. Let β stand for banned clique number of nodes. Then the current ARACNE has $\beta = 3$. We are suggesting that a β of 4 might be more likely to occur in Nature.

Chapter 3

Backdoor Adjustment Formula

The backdoor (BD) adjustment formula is proven in Chapter 17 from the rules of Do Calculus. The goal of this chapter is to give examples of the use of that theorem. We will restate the theorem in this chapter, sans proof. There is no need to understand the theorem's proof in order to use it. However, you will need to skim Chapter 17 in order to familiarize yourself with the notation used to state the theorem. This chapter also assumes that you are comfortable with the rules for checking for d-separation. Those rules are covered in Chapter 19.

Suppose that we have access to data that allows us to estimate a probability distribution $P(x., y., z.)$. Hence, the variables $\underline{x}.$, $\underline{y}.$, $\underline{z}.$ are ALL observed (i.e, not hidden). Then we say that the backdoor $\underline{z}.$ satisfies the **backdoor adjustment criterion** relative to $(\underline{x}., \underline{y}.)$ if

1. All backdoor paths from $\underline{x}.$ to $\underline{y}.$ are blocked by conditioning on $\underline{z}..$
2. $\underline{z}.. \cap de(\underline{x}.) = \emptyset$.

Claim 13 Backdoor Adjustment Formula

If $\underline{z}.$ satisfies the backdoor criterion relative to $(\underline{x}., \underline{y}.)$, then

$$P(y.|D\underline{x}. = x.) = \sum_{z.} P(y.|x., z.)P(z.) \quad (3.1)$$

$$= \sum_{z.} \begin{array}{c} z. \\ \searrow \\ x. \longrightarrow y. \end{array} \quad (3.2)$$

where $\sum z.$ means node $\underline{z}.$ is summed over.

proof: See Chapter 17.

QED

3.1 Examples

1.



BD criterion satisfied if $\underline{x}_\cdot = \underline{x}$, $\underline{y}_\cdot = \underline{y}$, $\underline{z}_\cdot = \emptyset$. No adjustment necessary.

$$P(y|\mathcal{D}\underline{x} = x) = P(y|x) \quad (3.4)$$

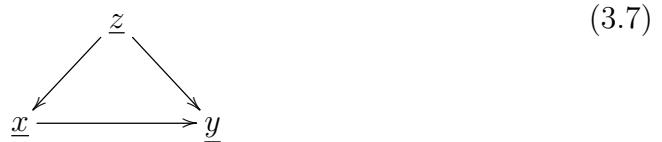
2.



BD criterion satisfied if $\underline{x}_\cdot = \underline{x}$, $\underline{y}_\cdot = \underline{y}$, $\underline{z}_\cdot = \emptyset$. No adjustment necessary.

$$P(y|\mathcal{D}\underline{x} = x) = P(y|x) \quad (3.6)$$

3.



BD criterion satisfied if $\underline{x}_\cdot = \underline{x}$, $\underline{y}_\cdot = \underline{y}$, $\underline{z}_\cdot = \underline{z}$.

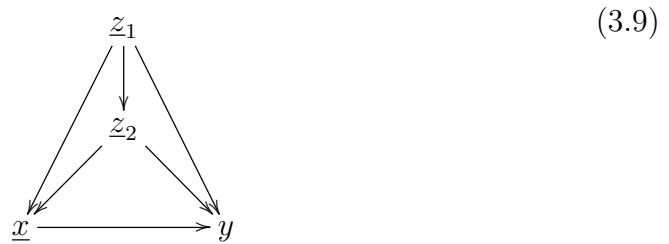
Note that here the backdoor formula adjusts the parents of x_\cdot .

4.



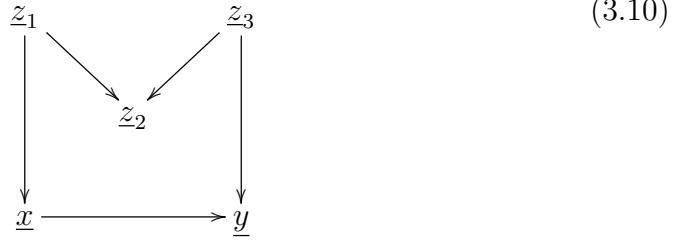
BD criterion satisfied if $\underline{x}_\cdot = \underline{x}$, $\underline{y}_\cdot = \underline{y}$, $\underline{z}_\cdot = \underline{z}$.

5.



BD criterion satisfied if $\underline{x}_. = \underline{x}, \underline{y}_. = \underline{y}, \underline{z}_. = (\underline{z}_1, \underline{z}_2)$.

6.



BD criterion satisfied if $\underline{x}_. = \underline{x}, \underline{y}_. = \underline{y}$ and $\underline{z}_.$ = one of the following.

- \emptyset
 - \underline{z}_3
 - $\underline{z}_2, \underline{z}_3$
 - $\underline{z}_1, \underline{z}_2, \underline{z}_3$
 - \underline{z}_1
 - $\underline{z}_1, \underline{z}_2$
 - $\underline{z}_1, \underline{z}_3$
-

7.



BD criterion is impossible to satisfy if $\underline{x}_. = \underline{x}, \underline{y}_. = \underline{y}$. However, the frontdoor criterion can be satisfied. See Chapter 25.

8.



BD criterion satisfied if $\underline{x}_. = \underline{x}, \underline{y}_. = \underline{y}, \underline{z}_. = \underline{z}$. We are able to block the backdoor path by conditioning on \underline{z} .

9.



Conditioning on \underline{z} blocks backdoor path $\underline{x}-\underline{z}-\underline{y}$, but opens path $\underline{x}-\underline{e}-\underline{z}-\underline{a}-\underline{y}$ because \underline{z} is a collider for that path. That path is blocked if we also condition on \underline{a} , which is possible because \underline{a} is observed. In conclusion, the BD criterion is satisfied if $\underline{x}_. = \underline{x}, \underline{y}_. = \underline{y}$ and $\underline{z}_. = (\underline{z}, \underline{a})$.

Conditioning on the parents of $\underline{x}.$ is often enough to block all backdoor paths. However, sometimes some of the parents are unobserved and one must condition on other nodes that are not parents of $\underline{x}.$ in order to satisfy the BD criterion.

10.



No need to control anything because only possible backdoor path is blocked by not conditioning on collider $\underline{w}.$ Hence,

$$P(y|\mathcal{D}\underline{x} = x) = P(y|x) . \quad (3.15)$$

However, if for some reason we want to control $\underline{t},$ we can do so. We can't control \underline{w} though, because $\underline{w} \in de(\underline{x}).$ Thus, the BD criterion is satisfied if $\underline{x} = \underline{x}, \underline{y} = \underline{y}$ and $\underline{z} = \underline{t}.$ Therefore,

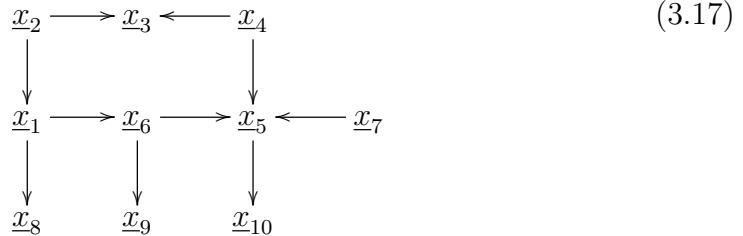
$$P(y|\mathcal{D}\underline{x} = x) = \sum_t P(y|x, t)P(t) . \quad (3.16)$$

11. Discuss what to do if several sets $\underline{z}.$ satisfy the BD criterion.

- Can evaluate $P(y|\mathcal{D}\underline{x} = x.)$ multiple ways and compare the results. This is a test that the causal bnet is correct.
 - It might be easier or less expensive to get data for some $\underline{z}.$ more than for others.
-

12. (Taken from online course notes Ref.[15])

Consider the bnet



If $\underline{x} = \underline{x}_1$ and $\underline{y} = \underline{x}_5,$ find all possible adjustment multinode $\underline{z}.$ that satisfy the BD criterion. Ans:

- \emptyset
- \underline{x}_4
- $\underline{x}_2, \underline{x}_3$
- $\underline{x}_2, \underline{x}_3, \underline{x}_4$
- \underline{x}_2
- $\underline{x}_2, \underline{x}_4$
- $\underline{x}_3, \underline{x}_4$

Add \underline{x}_7 to each of the previous 7 possible $\underline{z}..$. This gives a total of 14 possible adjustment multinodes $\underline{z}..$.

Chapter 4

Back Propagation (Automatic Differentiation)

4.1 General Theory

4.1.1 Jacobians

Suppose $f : \mathbb{R}^{nx} \rightarrow \mathbb{R}^{nf}$ and

$$y = f(x) . \quad (4.1)$$

Then the Jacobian $\frac{\partial y}{\partial x}$ is defined as the matrix with entries¹

$$\left[\frac{\partial y}{\partial x} \right]_{i,j} = \frac{\partial y_i}{\partial x_j} . \quad (4.2)$$

Jacobian of function composition. Suppose $f : \mathbb{R}^{nx} \rightarrow \mathbb{R}^{nf}$, $g : \mathbb{R}^{nf} \rightarrow \mathbb{R}^{ng}$. If

$$y = g \circ f(x) , \quad (4.3)$$

then

$$\frac{\partial y}{\partial x} = \frac{\partial g}{\partial f} \frac{\partial f}{\partial x} . \quad (4.4)$$

Right hand side of last equation is a product of two matrices so order of matrices is important.

$$\underline{f}^4 \longleftarrow \underline{f}^3 \longleftarrow \underline{f}^2 \longleftarrow \underline{f}^1 \longleftarrow \underline{f}^0$$

(a) Composition

$$\frac{\partial f^4}{\partial x} \longleftarrow \frac{\partial f^3}{\partial x} \longleftarrow \frac{\partial f^2}{\partial x} \longleftarrow \frac{\partial f^1}{\partial x} \longleftarrow 1$$

(b) Forward-p

$$1 \longrightarrow \frac{\partial y}{\partial f^3} \longrightarrow \frac{\partial y}{\partial f^2} \longrightarrow \frac{\partial y}{\partial f^1} \longrightarrow \frac{\partial y}{\partial f^0}$$

(c) Back-p

Figure 4.1: bnets for function composition, forward propagation and back propagation for $nf = 5$ nodes.

4.1.2 Bnets for function composition, forward propagation and back propagation

Let

$$y = f^4 \circ f^3 \circ f^2 \circ f^1(x) . \quad (4.5)$$

This function composition chain can be represented by the bnet Fig.4.1(a) with TPMs

$$P(f^\mu | f^{\mu-1}) = \mathbb{1}(f^\mu = f^\mu(f^{\mu-1})) \quad (4.6)$$

for $\mu = 1, 2, 3, 4$.

¹Mnemonic for remembering order of indices: i in numerator/ j in denominator becomes index i/j of Jacobian matrix.

Note that

$$\frac{\partial y}{\partial x} = \frac{\partial y}{\partial f^3} \frac{\partial f^3}{\partial f^2} \left[\frac{\partial f^2}{\partial f^1} \frac{\partial f^1}{\partial x} \right] \quad (4.7)$$

$$= \frac{\partial y}{\partial f^3} \left[\frac{\partial f^3}{\partial f^2} \frac{\partial f^2}{\partial x} \right] \quad (4.8)$$

$$= \left[\frac{\partial y}{\partial f^3} \frac{\partial f^3}{\partial x} \right] \quad (4.9)$$

$$= \frac{\partial y}{\partial x}. \quad (4.10)$$

This forward propagation can be represented by the bnet Fig.4.1(b) with node TPMs

$$P\left(\frac{\partial f^{\mu+1}}{\partial x} \mid \frac{\partial f^\mu}{\partial x}\right) = \mathbb{1}\left(\frac{\partial f^{\mu+1}}{\partial x} = \frac{\partial f^{\mu+1}}{\partial f^\mu} \frac{\partial f^\mu}{\partial x}\right) \quad (4.11)$$

for $\mu = 1, 2, 3$.

Note that

$$\frac{\partial y}{\partial x} = \left[\frac{\partial y}{\partial f^3} \frac{\partial f^3}{\partial f^2} \right] \frac{\partial f^2}{\partial f^1} \frac{\partial f^1}{\partial x} \quad (4.12)$$

$$= \left[\frac{\partial y}{\partial f^2} \frac{\partial f^2}{\partial f^1} \right] \frac{\partial f^1}{\partial x} \quad (4.13)$$

$$= \left[\frac{\partial y}{\partial f^1} \frac{\partial f^1}{\partial x} \right] \quad (4.14)$$

$$= \frac{\partial y}{\partial x}. \quad (4.15)$$

This back propagation can be represented by the bnet Fig.4.1(c) with node TPMs

$$P\left(\frac{\partial y}{\partial f^\mu} \mid \frac{\partial y}{\partial f^{\mu+1}}\right) = \mathbb{1}\left(\frac{\partial y}{\partial f^\mu} = \frac{\partial y}{\partial f^{\mu+1}} \frac{\partial f^{\mu+1}}{\partial f^\mu}\right) \quad (4.16)$$

for $\mu = 2, 1, 0$.

$\frac{\partial f^{\mu+1}}{\partial f^\mu}$ is a Jacobian matrix so the order of multiplication matters. In forward prop, it pre-multiplies, and in back prop it post-multiplies.

4.2 Application to Neural Networks

4.2.1 Absorbing b_i^λ into $w_{i|j}$.

The TPMs, printed in blue, for a NN bnet, as given in Chapter 50, are as follows.

For all hidden layers $\lambda = 0, 1, \dots, \Lambda - 2$,

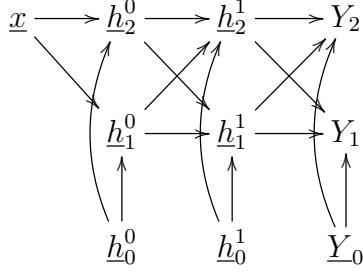


Figure 4.2: Nodes $\underline{h}_0^0, \underline{h}_0^1, \underline{Y}_0$ are all set to 1. They allow us to absorb b_i^λ into the first column of $w_{i|j}^\lambda$.

$$P(h_i^\lambda | h_{\cdot}^{\lambda-1}) = \delta \left(h_i^\lambda, \mathcal{A}_i^\lambda \left(\sum_j w_{i|j}^\lambda h_j^{\lambda-1} + b_i^\lambda \right) \right) \quad (4.17)$$

for $i = 0, 1, \dots, nh(\lambda) - 1$. For the output visible layer $\lambda = \Lambda - 1$:

$$P(Y_i | h_{\cdot}^{\Lambda-2}) = \delta \left(Y_i, \mathcal{A}_i^{\Lambda-1} \left(\sum_j w_{i|j}^{\Lambda-1} h_j^{\Lambda-2} + b_i^{\Lambda-1} \right) \right) \quad (4.18)$$

for $i = 0, 1, \dots, ny - 1$.

For each λ , replace the matrix $w_{\cdot|}^\lambda$ by the augmented matrix $[b_{\cdot|}^\lambda, w_{\cdot|}^\lambda]$ so that the new $w_{\cdot|}^\lambda$ satisfies

$$w_{i|0}^\lambda = b_i^\lambda \quad (4.19)$$

Let the nodes \underline{h}_0^λ for all λ and \underline{Y}_0 be root nodes (so no arrows pointing into them). For each λ , draw arrows from \underline{h}_0^λ to all other nodes in that same layer. Draw arrows from \underline{Y}_0 to all other nodes in that same layer.

After performing the above steps, the TPMs, printed in blue, for the NN bnet, are as follows:

For all hidden layers $\lambda = 0, 1, \dots, \Lambda - 2$,

$$P(h_0^\lambda) = \delta(h_0^\lambda, 1) , \quad (4.20)$$

and

$$P(h_i^\lambda | h_{\cdot}^{\lambda-1}, h_0^\lambda = 1) = \delta \left(h_i^\lambda, \mathcal{A}_i^\lambda \left(\sum_j w_{i|j}^\lambda h_j^{\lambda-1} \right) \right) \quad (4.21)$$

for $i = 1, \dots, nh(\lambda) - 1$. For the output visible layer $\lambda = \Lambda - 1$:

$$P(Y_0) = \delta(Y_0, 1) , \quad (4.22)$$

and

$$P(Y_i \mid h_{\cdot}^{\Lambda-2}, Y_0 = 1) = \delta \left(Y_i, \mathcal{A}_i^{\Lambda-1} \left(\sum_j w_{i|j}^{\Lambda-1} h_j^{\Lambda-2} \right) \right) \quad (4.23)$$

for $i = 1, 2, \dots, ny - 1$.

4.2.2 Bnets for function composition, forward propagation and back propagation for NN

$$\underline{\mathcal{A}}^3 \longleftarrow \underline{\mathcal{B}}^3 \longleftarrow \underline{\mathcal{A}}^2 \longleftarrow \underline{\mathcal{B}}^2 \longleftarrow \underline{\mathcal{A}}^1 \longleftarrow \underline{\mathcal{B}}^1 \longleftarrow \underline{\mathcal{A}}^0 \longleftarrow \underline{\mathcal{B}}^0 \longleftarrow \underline{x}$$

(a)

$$\underline{\frac{\partial \mathcal{A}^3}{\partial x}} \longleftarrow \underline{\frac{\partial \mathcal{B}^3}{\partial x}} \longleftarrow \underline{\frac{\partial \mathcal{A}^2}{\partial x}} \longleftarrow \underline{\frac{\partial \mathcal{B}^2}{\partial x}} \longleftarrow \underline{\frac{\partial \mathcal{A}^1}{\partial x}} \longleftarrow \underline{\frac{\partial \mathcal{B}^1}{\partial x}} \longleftarrow \underline{\frac{\partial \mathcal{A}^0}{\partial x}} \longleftarrow \underline{\frac{\partial \mathcal{B}^0}{\partial x}} \longleftarrow \underline{1}$$

(b)

$$\underline{1} \longrightarrow \underline{\frac{\partial Y}{\partial \mathcal{B}^3}} \longrightarrow \underline{\frac{\partial Y}{\partial \mathcal{A}^2}} \longrightarrow \underline{\frac{\partial Y}{\partial \mathcal{B}^2}} \longrightarrow \underline{\frac{\partial Y}{\partial \mathcal{A}^1}} \longrightarrow \underline{\frac{\partial Y}{\partial \mathcal{B}^1}} \longrightarrow \underline{\frac{\partial Y}{\partial \mathcal{A}^0}} \longrightarrow \underline{\frac{\partial Y}{\partial \mathcal{B}^0}} \longrightarrow \underline{\frac{\partial Y}{\partial x}}$$

(c)

Figure 4.3: bnets for (a) function composition, (b) forward propagation and (c) back propagation for a neural net with 4 layers (3 hidden and output visible).

From here on, we will rename y above by $Y = \hat{y}$ and consider samples $y[i]$ for $i = 0, 1, \dots, nsam - 1$. The Error (aka loss or cost function) is

$$\mathcal{E} = \frac{1}{nsam} \sum_{\sigma=0}^{nsam-1} \sum_{i=0}^{ny-1} |Y_i - y_i[\sigma]|^2 \quad (4.24)$$

To perform simple gradient descent, one uses:

$$(w_{i|j}^\lambda)' = w_{i|j}^\lambda - \eta \frac{\partial \mathcal{E}}{\partial w_{i|j}^\lambda} . \quad (4.25)$$

One has

$$\frac{\partial \mathcal{E}}{\partial w_{i|j}^\lambda} = \frac{1}{nsam} \sum_{\sigma=0}^{nsam-1} \sum_{i=0}^{ny-1} 2(Y_i - y_i[\sigma]) \frac{\partial Y}{\partial w_{i|j}^\lambda}. \quad (4.26)$$

Define \mathcal{B}_i^λ thus

$$\mathcal{B}_i^\lambda(h^{\lambda-1}) = \sum_j w_{i|j}^\lambda h_j^{\lambda-1}. \quad (4.27)$$

Then

$$\frac{\partial Y}{\partial w_{i|j}^\lambda} = \frac{\partial Y}{\partial \mathcal{B}_i^\lambda} \frac{\partial \mathcal{B}_i^\lambda}{\partial w_{i|j}^\lambda} \quad (4.28)$$

$$= \frac{\partial Y}{\partial \mathcal{B}_i^\lambda} h_j^{\lambda-1} \quad (4.29)$$

$$\frac{\partial \mathcal{E}}{\partial w_{i|j}^\lambda} = \frac{\partial \mathcal{E}}{\partial \mathcal{B}_j^\lambda} \frac{\partial \mathcal{B}_j^\lambda}{\partial w_{i|j}^\lambda} \quad (4.30)$$

$$= \frac{\partial \mathcal{E}}{\partial \mathcal{B}_j^\lambda} h_j^{\lambda-1}. \quad (4.31)$$

This suggest that we can calculate the derivatives of the error \mathcal{E} with respect to the weights $w_{i|j}^\lambda$ in two stages, using an intermediate quantity δ_j^λ :

$$\begin{cases} \delta_j^\lambda = \frac{\partial \mathcal{E}}{\partial \mathcal{B}_j^\lambda} \\ \frac{\partial \mathcal{E}}{\partial w_{i|j}^\lambda} = \delta_j^\lambda h_j^{\lambda-1} \end{cases} \quad (4.32)$$

To apply what we learned in the earlier General Theory section of this chapter, consider a NN with 4 layers (3 hidden, and the output visible one). Define the functions f_i as follows:

$$f_i^0 = x_i \quad (4.33)$$

$$\text{Layer 0: } f_i^1 = \mathcal{B}_i^0(x_i), \quad f_i^2 = \mathcal{A}_i^0(\mathcal{B}_i^0) \quad (4.34)$$

$$\text{Layer 1: } f_i^3 = \mathcal{B}_i^1(\mathcal{A}_i^0), \quad f_i^4 = \mathcal{A}_i^1(\mathcal{B}_i^1) \quad (4.35)$$

$$\text{Layer 2: } f_i^5 = \mathcal{B}_i^2(\mathcal{A}_i^1), \quad f_i^6 = \mathcal{A}_i^2(\mathcal{B}_i^2) \quad (4.36)$$

$$\text{Layer 3: } f_i^7 = \mathcal{B}_i^3(\mathcal{A}_i^2), \quad f_i^8 = \mathcal{A}_i^3(\mathcal{B}_i^3) \quad (4.37)$$

See Fig.4.3. The TPMs, printed in blue, for the bnet (c) for back propagation, are as follows:

$$P\left(\frac{\partial Y}{\partial \mathcal{B}^\lambda} \mid \frac{\partial Y}{\partial \mathcal{B}^{\lambda+1}}\right) = \mathbb{1}\left(\frac{\partial Y}{\partial \mathcal{B}^\lambda} = \frac{\partial Y}{\partial \mathcal{B}^{\lambda+1}} \frac{\partial \mathcal{B}^{\lambda+1}}{\partial \mathcal{A}^\lambda} \frac{\partial \mathcal{A}^\lambda}{\partial \mathcal{B}^\lambda}\right). \quad (4.38)$$

One has

$$\frac{\partial \mathcal{A}_i^\lambda}{\partial \mathcal{B}_j^\lambda} = D\mathcal{A}_i^\lambda(\mathcal{B}_i^\lambda) \delta(i, j) \quad (4.39)$$

where $D\mathcal{A}_i^\lambda(z)$ is the derivative of $\mathcal{A}_i^\lambda(z)$.

From Eq.(4.27)

$$\mathcal{B}_i^{\lambda+1}(\mathcal{A}^\lambda) = \sum_j w_{i|j}^{\lambda+1} \mathcal{A}_j^\lambda \quad (4.40)$$

so

$$\frac{\partial \mathcal{B}_i^{\lambda+1}}{\partial \mathcal{A}_j^\lambda} = w_{i|j}^{\lambda+1}. \quad (4.41)$$

Therefore, Eq.(4.38) implies

$$P\left(\frac{\partial Y}{\partial \mathcal{B}_j^\lambda} \mid \frac{\partial Y}{\partial \mathcal{B}_j^{\lambda+1}}\right) = \mathbb{1}\left(\frac{\partial Y}{\partial \mathcal{B}_j^\lambda} = \sum_i \frac{\partial Y}{\partial \mathcal{B}_i^{\lambda+1}} D\mathcal{A}_j^\lambda(\mathcal{B}_j^\lambda) w_{i|j}^{\lambda+1}\right), \quad (4.42)$$

$$P\left(\frac{\partial \mathcal{E}}{\partial \mathcal{B}_j^\lambda} \mid \frac{\partial \mathcal{E}}{\partial \mathcal{B}_j^{\lambda+1}}\right) = \mathbb{1}\left(\frac{\partial \mathcal{E}}{\partial \mathcal{B}_j^\lambda} = \sum_i \frac{\partial \mathcal{E}}{\partial \mathcal{B}_i^{\lambda+1}} D\mathcal{A}_j^\lambda(\mathcal{B}_j^\lambda) w_{i|j}^{\lambda+1}\right), \quad (4.43)$$

$$P(\delta_j^\lambda \mid \delta_j^{\lambda+1}) = \mathbb{1}\left(\delta_j^\lambda = \sum_i \delta_i^{\lambda+1} D\mathcal{A}_j^\lambda(\mathcal{B}_j^\lambda) w_{i|j}^{\lambda+1}\right). \quad (4.44)$$

First delta of iteration, belonging to output layer $\lambda = \Lambda - 1$:

$$\delta_j^{\Lambda-1} = \frac{\partial \mathcal{E}}{\partial \mathcal{B}_j^{\Lambda-1}} \quad (4.45)$$

$$= \frac{1}{nsam} \sum_{\sigma=0}^{nsam-1} \sum_{i=0}^{ny-1} 2(Y_i - y_i[\sigma]) D\mathcal{A}_i^{\Lambda-1}(\mathcal{B}_i^{\Lambda-1}) \delta(i, j) \quad (4.46)$$

$$= \frac{1}{nsam} \sum_{\sigma=0}^{nsam-1} 2(Y_j - y_j[\sigma]) D\mathcal{A}_j^{\Lambda-1}(\mathcal{B}_j^{\Lambda-1}) \quad (4.47)$$

Cute expression for derivative of sigmoid function:

$$Dsmoid(x) = smoid(x)(1 - smoid(x)) \quad (4.48)$$

4.3 General bnets instead of Markov chains induced by layered structure of NNs

$$P(\delta_{\underline{x}} \mid (\delta_{\underline{a}})_{\underline{a} \in ch(\underline{x})}) = \mathbb{1}(\delta_{\underline{x}} = \sum_{\underline{a} \in ch(\underline{x})} \delta_{\underline{a}} D\mathcal{A}_{\underline{x}}(\mathcal{B}_{\underline{x}})) w_{\underline{a}|\underline{x}}) \quad (4.49)$$

Reverse arrows of original bnet and define the TPM of nodes of “time reversed” bnet by

$$P(\delta_{\underline{x}} \mid (\delta_{\underline{a}})_{\underline{a} \in pa(\underline{x})}) = \mathbb{1}(\delta_{\underline{x}} = \sum_{\underline{a} \in pa(\underline{x})} \delta_{\underline{a}} D\mathcal{A}_{\underline{x}}(\mathcal{B}_{\underline{x}})) w_{\underline{x}|\underline{a}}^T) \quad (4.50)$$

Chapter 5

Basic Curve Fitting Using Gradient Descent

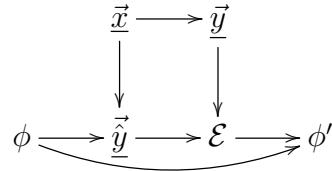


Figure 5.1: Basic curve fitting bnet.

Samples $(x^\sigma, y^\sigma) \in S_{\underline{x}} \times S_{\underline{y}}$ are given. $nsam(\vec{x}) = nsam(\vec{y})$. Estimator function $\hat{y}(x; \phi)$ for $x \in S_{\underline{x}}$ and $\phi \in \mathbb{R}$ is given.

Let

$$P_{\underline{x}, \underline{y}}(x, y) = \frac{1}{nsam(\vec{x})} \sum_{\sigma} \mathbb{1}(x = x^\sigma, y = y^\sigma). \quad (5.1)$$

Let

$$\mathcal{E}(\vec{x}, \vec{y}, \phi) = \frac{1}{nsam(\vec{y})} \sum_{\sigma} |y^\sigma - \hat{y}(x^\sigma; \phi)|^2 \quad (5.2)$$

\mathcal{E} is called the mean square error.

Best fit is parameters ϕ^* such that

$$\phi^* = \operatorname{argmin}_{\phi} \mathcal{E}(\vec{x}, \vec{y}, \phi). \quad (5.3)$$

The TPMs, printed in blue, for the basic curve fitting bnet Fig.5.1, are as follows.

$$P(\phi) = \text{given}. \quad (5.4)$$

The first time it is used, ϕ is arbitrary. After the first time, it is determined by previous stage.

$$P(\vec{x}) = \prod_{\sigma} P_{\underline{x}}(x^{\sigma}) \quad (5.5)$$

$$P(\vec{y}|\vec{x}) = \prod_{\sigma} P_{\underline{y}|\underline{x}}(y^{\sigma} | x^{\sigma}) \quad (5.6)$$

$$P(\hat{y}^{\sigma}|\phi, \vec{x}) = \delta(\hat{y}^{\sigma}, \hat{y}(x^{\sigma}; \phi)) \quad (5.7)$$

$$P(\mathcal{E}|\vec{y}, \vec{y}) = \delta(\mathcal{E}, \frac{1}{nsam(\vec{x})} \sum_{\sigma} |y^{\sigma} - \hat{y}^{\sigma}|^2) . \quad (5.8)$$

$$P(\phi'|\phi, \mathcal{E}) = \delta(\phi', \phi - \eta \partial_{\phi} \mathcal{E}) \quad (5.9)$$

$\eta > 0$ is the descent rate. If $\Delta\phi = \phi' - \phi = -\eta \frac{\partial \mathcal{E}}{\partial \phi}$, then $\Delta\mathcal{E} = \frac{-1}{\eta} (\Delta\phi)^2 < 0$ so this will minimize the error \mathcal{E} . This is called “gradient descent”.

Chapter 6

Bell and Clauser-Horne Inequalities in Quantum Mechanics

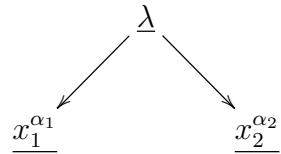


Figure 6.1: bnet used to discuss Bell and Clauser-Horne inequalities in Quantum Mechanics.

I wrote an article about this in 2008 for my blog “Quantum Bayesian Networks”. See Ref.[58].

Chapter 7

Berkson's Paradox

For more information about Berkson's Paradox (BP), see Ref.[70]

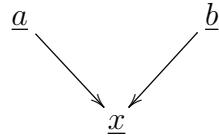


Figure 7.1: Bnet used to discuss Berkson's Paradox (BP). \underline{a} and \underline{b} are both causes of collider \underline{x} .

Consider the bnet of Fig.7.1. For that bnet, we have

$$P(a, b, x) = P(a)P(b)P(x|a, b) . \quad (7.1)$$

Summing Eq.(7.1) over x , we get

$$P(a, b) = P(a)P(b) \quad (7.2)$$

so \underline{a} and \underline{b} are independent. It follows that a can be ignored in calculating the probability of b ; i.e.,

$$\boxed{P(b|a) = P(b)} . \quad (7.3)$$

However, a cannot be ignored in calculating the probability of b , if x is being held fixed; i.e.,

$$\boxed{P(b|a, x) \neq P(b|x)} . \quad (7.4)$$

Indeed,

$$P(b|a, x) = \frac{P(b)P(x|a, b)}{\sum_b P(b)P(x|a, b)} \quad (7.5)$$

whereas

$$P(b|x) = \frac{\sum_a P(a)P(b)P(x|a,b)}{\sum_{a,b} P(a)P(b)P(x|a,b)} . \quad (7.6)$$

The two boxed equations are what is referred to as BP.

BP is also called **collider bias** because x is a collider.

BP is also called **explaining away** in the special case that $a, b, x \in \{true, false\} = \{0, 1\}$. In that case, if x is fixed to true, and the cause a is known to be true, then the cause b is less likely to be true. For example, suppose a car engine fails ($x = 1$) and the two most likely causes of the failure are alternator (a) and battery (b). Once we know that the alternator has failed ($a = 1$), it is less likely that the battery is failing ($b = 1$) than when the status of a was not known; i.e., $P(b = 1|x = 1, a = 1) < P(b = 1|x = 1)$.

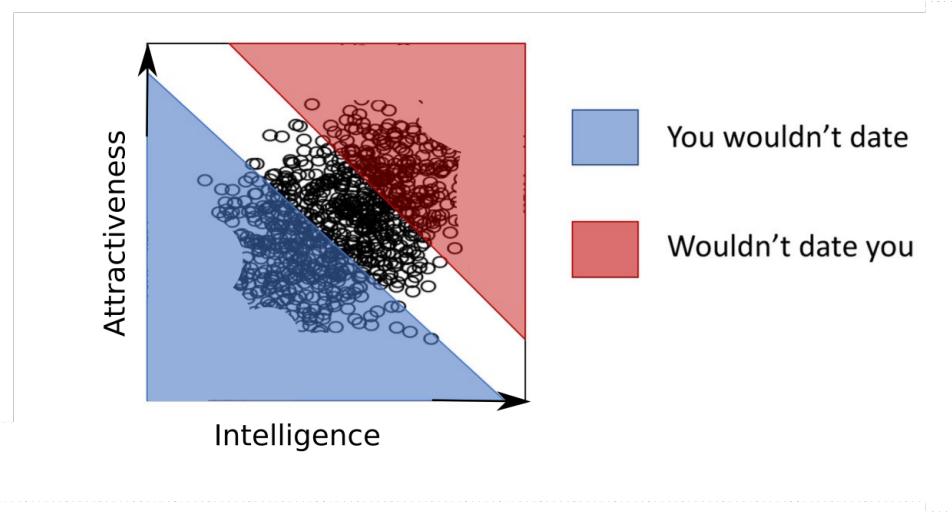


Figure 7.2: Example of Berkson's paradox (BP).

Fig.7.2 presents an example of BP. The figure consists of a scatter plot with axes $x=$ intelligence, $y=$ attractiveness, for a population of possible dates for a single person. For the full population,

$$(a, b) \sim P(a, b) = P(a)P(b) \quad (7.7)$$

whereas for the population in the white swath,

$$(a, b) \sim P(a, b|x) = P(b|a, x)P(a|x) \neq P(b|x)P(a|x) . \quad (7.8)$$

As shown by Fig.7.2, BP is an example of **selection bias**. Selection bias happens when a non-representative subset of the total population is considered (i.e., selected).

Chapter 8

Binary Decision Diagrams

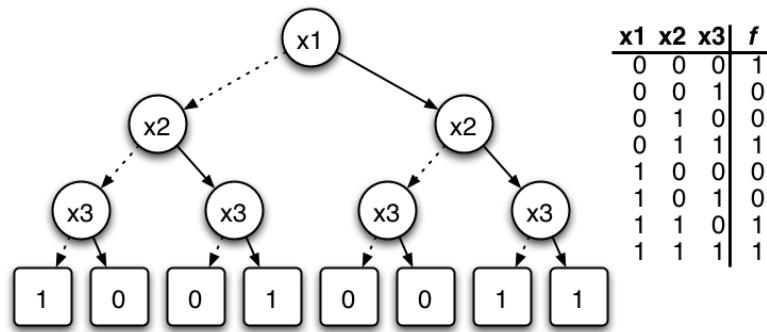


Figure 8.1: Binary decision tree and truth table for the function $f(x_1, x_2, x_3) = \bar{x}_1(x_2 + \bar{x}_3) + x_1x_2$

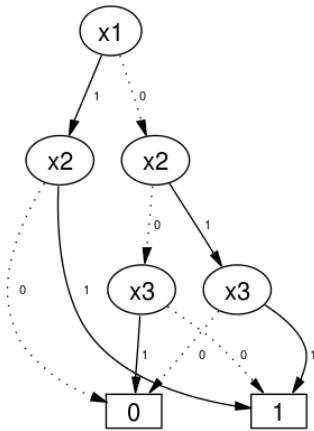


Figure 8.2: BDD for the function f of Fig.8.1.

This chapter is based on Wikipedia article Ref.[74].

Binary Decision Diagrams (BDDs) can be understood as a special case of Decision Trees (dtrees). We will assume that the reader has read Chapter 13 on dtrees before reading this chapter.

Both Figs.8.1 and 8.2 were taken from the aforementioned Wikipedia article. They give a simple example of a function $f : \{0, 1\}^3 \rightarrow \{0, 1\}$ represented in Fig.8.1 as a **binary decision tree** and in Fig.8.2 as a **binary decision diagram (BDD)**. It is possible to find, for each of those two figures, a bnet with the same graph structure. We show how to do this next.

We begin by noting that the function $f : \{0, 1\}^3 \rightarrow \{0, 1\}$ is a special case of a probability distribution $P : \{0, 1\}^3 \rightarrow [0, 1]$. In fact, if we restrict P to be deterministic, then $P_{det} : \{0, 1\}^3 \rightarrow \{0, 1\}$ has the same domain and range as f . Henceforth, we will refer to $f(x_1, x_2, x_3)$ as $P(x_1, x_2, x_3)$, keeping in mind that we are restricting our attention to deterministic probability distributions.

If we apply the chain rule for conditional probabilities to $P(x_1, x_2, x_3)$, we get

$$P(x_1, x_2, x_3) = P(x_3|x_1, x_2)P(x_2|x_1)P(x_1), \quad (8.1)$$

which can be represented by the bnet:

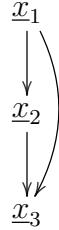


Figure 8.3: Most general 3 node bnet.

But in Chapter 13, we learned how to represent the dtree of Fig.8.1 as the image bnet Fig.8.4. The TPMs, printed in blue, for the image bnet Fig.8.4, are as follows. Note that the TPMs for Fig.8.4 can be constructed from the TPMs for the bnet Fig.8.3. If $x_1, x_2, x_3, x_4 \in \{0, 1, null\}$ and $a, b, c \in \{0, 1\}$, then

$$P(\underline{x}_1 = x_1) = \begin{cases} P_{x_1}(x_1) & \text{if } x_1 \in \{0, 1\} \\ 0 & \text{if } x_1 = \text{null} \end{cases} \quad (8.2)$$

$$P(\underline{x}_2|a = x_2 | \underline{x}_1 = x_1) = \begin{cases} P_{x_2|x_1}(x_2|a) & \text{if } x_1 = a \\ \mathbb{1}(x_2 = \text{null}) & \text{otherwise} \end{cases} \quad (8.3)$$

$$P(\underline{x}_3|a, b = x_3 | \underline{x}_2|a = x_2) = \begin{cases} P_{x_3|x_1, x_2}(x_3|a, b) & \text{if } x_2 = b \\ \mathbb{1}(x_3 = \text{null}) & \text{otherwise} \end{cases} \quad (8.4)$$

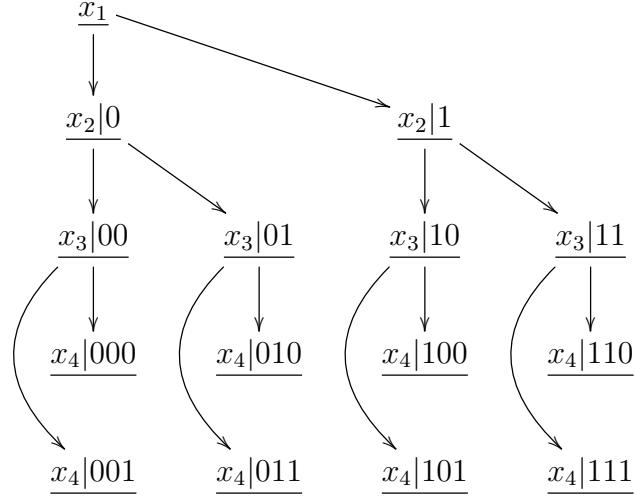


Figure 8.4: Image bnet for binary dtree of Fig.8.1.

$$P(\underline{x}_4|a, b, c = x_4 | \underline{x}_3|a, b = x_3) = \begin{cases} \delta(x_4, c) & \text{if } x_3 = c \\ \mathbb{1}(x_4 = \text{null}) & \text{otherwise} \end{cases} \quad (8.5)$$

Note that if $P_{\underline{x}_3|x_1, x_2} = P_{\underline{x}_3|x_2}$ in Eq.(8.4), then the bnet Fig.8.3 reduces to a Markov chain $\underline{x}_1 \rightarrow \underline{x}_2 \rightarrow \underline{x}_3$.

The BDD shown in Fig.8.2 emphasizes the fact that

$$P(x_1, x_2, x_3|x_1 = 1) = P(x_2|x_1 = 1) = x_2 . \quad (8.6)$$

The BDD of Fig.8.2 has as image bnet Fig.8.5. Define

$$pa(\underline{0}) = pa(\underline{1}) = (x_2|1, x_3|00, x_3|01) . \quad (8.7)$$

Let $pa(\underline{0}) = abc$ mean the same as $pa(\underline{0}) = (a, b, c)$. The TPMs of the image bnet Fig.8.5 are the same as those for the image bnet Fig.8.4 except for the TPMs of the nodes $\underline{0}$ and $\underline{1}$. For those two nodes, the TPMs, printed in blue, are as follows.

$$P(\underline{0} = x|pa(\underline{0})) = \begin{cases} \delta(x, 0) & \text{if } pa(\underline{0}) = 011 \\ \delta(x, \text{null}) & \text{otherwise} \end{cases} \quad (8.8)$$

$$P(\underline{1} = x|pa(\underline{1})) = \begin{cases} \delta(x, 1) & \text{if } pa(\underline{1}) = 101 \\ \delta(x, \text{null}) & \text{otherwise} \end{cases} \quad (8.9)$$

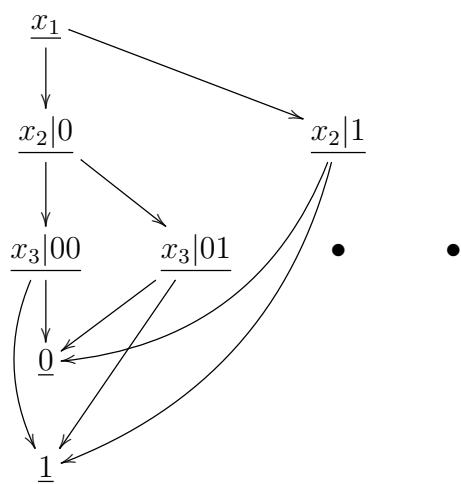


Figure 8.5: Image bnet for BDD of Fig.8.2.

Chapter 9

Chow-Liu Trees and Tree Augmented Naive Bayes (TAN)

This chapter is mostly based on chapter 8 of Pearl's 1988 book Ref.[37]. See also Ref.[79] and references therein.

This chapter uses various Shannon Information Theory entropies. Our notation for these entropies is described in Chapter Notational Conventions and Preliminaries.

9.1 Chow-Liu Trees

Chow-Liu trees refers to an algorithm for finding a bnet tree that fits an a priori given probability distribution as closely as possible.

Consider a bnet with n nodes $\underline{x}^n = (\underline{x}_0, \underline{x}_1, \dots, \underline{x}_{n-1})$ such that $\underline{x}_i \in S_{\underline{x}_i}$ for all i . Let its total probability distribution be $P_{\underline{x}^n}$. For simplicity, we will abbreviate $P_{\underline{x}^n}$ by P . Hence

$$P(x^n) = P_{\underline{x}^n}(x^n). \quad (9.1)$$

Suppose we want to fit $P_{\underline{x}^n}$ by a tree bnet with nodes $\underline{t}^n = (\underline{t}_0, \underline{t}_1, \dots, \underline{t}_{n-1})$ such that $\underline{t}_i \in S_{\underline{t}_i} = S_{\underline{x}_i}$ for all i . For simplicity, we will abbreviate $P_{\underline{t}^n}$ by P_T . Hence

$$P_T(x^n) = P_{\underline{t}^n}(x^n). \quad (9.2)$$

Throughout this chapter, let $V = \{0, 1, \dots, n-1\}$, the set of vertices. Suppose μ is a function $\mu : V \rightarrow V$ such that $\mu(i) < i$. Let $T_\mu = \{\underline{t}_{\mu(i)} \rightarrow \underline{t}_i : i \in V - \{0\}\}$. Then T_μ is a tree that spans (i.e., it includes all nodes) \underline{t}^n . Its root node is \underline{t}_0 , because \underline{t}_0 has no parents. All other nodes \underline{t}_i have exactly one parent, namely $\underline{t}_{\mu(i)}$. Let P_T , the total probability distribution for the tree, be parameterized by the function μ as follows:

$$P_T(x^n) = \prod_{i=0}^{n-1} P_T(x_i|x_{\mu(i)}) , \quad (9.3)$$

where, for the root node 0, $P_T(x_0|x_{\mu(0)}) = P_T(x_0)$.

Claim 14 $D_{KL}(P \parallel P_T)$ is minimized over all probability distributions P_T that are expressible as Eq.(9.3) iff

$$P_T(x_i|x_{\mu(i)}) = P(x_i|x_{\mu(i)}) \quad (9.4)$$

for all i , and

$$\sum_i H(\underline{x}_i : \underline{x}_{\mu(i)}) \quad (9.5)$$

is maximized over all μ .

proof:

$$D_{KL}(P \parallel P_T) = \sum_{x^n} P(x^n) \ln \frac{P(x^n)}{P_T(x^n)} \quad (9.6)$$

$$= - \sum_{x^n} \sum_i P(x^n) \ln P_T(x_i|x_{\mu(i)}) + \sum_i P(x^n) \ln P(x^n) \quad (9.7)$$

$$= - \sum_i \sum_{x_i, x_{\mu(i)}} P(x_i, x_{\mu(i)}) \ln P_T(x_i|x_{\mu(i)}) - H(\underline{x}^n) \quad (9.8)$$

$$= - \sum_i \underbrace{\sum_{x_{\mu(i)}} P(x_{\mu(i)}) \left[\sum_{x_i} P(x_i|x_{\mu(i)}) \ln P_T(x_i|x_{\mu(i)}) \right]}_{=H(\underline{x}_i|x_{\mu(i)})=H(\underline{x}_i:x_{\mu(i)})-H(\underline{x}_i)} - H(\underline{x}^n) . \quad (9.9)$$

Now note that

$$\sum_{x_i} P(x_i|x_{\mu(i)}) \ln \frac{P(x_i|x_{\mu(i)})}{P_T(x_i|x_{\mu(i)})} \geq 0 \quad (9.10)$$

and this inequality becomes an equality iff

$$P(x_i|x_{\mu(i)}) = P_T(x_i|x_{\mu(i)}) . \quad (9.11)$$

Therefore

$$D_{KL}(P \parallel P_T) \geq - \underbrace{\sum_i \sum_{x_{\mu(i)}} P(x_{\mu(i)}) \left[\sum_{x_i} P(x_i|x_{\mu(i)}) \ln P(x_i|x_{\mu(i)}) \right]}_{=H(\underline{x}_i|x_{\mu(i)})=H(\underline{x}_i:x_{\mu(i)})-H(\underline{x}_i)} - H(\underline{x}^n) , \quad (9.12)$$

and this inequality becomes an equality iff Eq.(9.11) is satisfied.

Note from the last equation that

$$\operatorname{argmin}_{\mu} D_{KL}(P \parallel P_T) = \operatorname{argmax}_{\mu} \sum_i H(\underline{x}_i : \underline{x}_{\mu(i)}) . \quad (9.13)$$

QED

Claim 15

$$\operatorname{argmin}_{\mu} H(\underline{x}^n) = \operatorname{argmax}_{\mu} \sum_i H(\underline{x}_i : \underline{x}_{\mu(i)}) \quad (9.14)$$

proof:

$$H(\underline{x}^n) = - \sum_{x^n} P(x^n) \sum_i \ln P(x_i | x_{\mu(i)}) \quad (9.15)$$

$$= - \sum_i \sum_{x_i, x_{\mu(i)}} P(x_i, x_{\mu(i)}) \ln P(x_i | x_{\mu(i)}) \quad (9.16)$$

$$= - \sum_i \sum_{x_i, x_{\mu(i)}} P(x_i, x_{\mu(i)}) \left[\ln \frac{P(x_i | x_{\mu(i)})}{P(x_i)} + \ln P(x_i) \right] \quad (9.17)$$

$$= - \sum_i [H(\underline{x}_i : \underline{x}_{\mu(i)}) - H(\underline{x}_i)] \quad (9.18)$$

$$= \sum_i H(\underline{x}_i) - \sum_i H(\underline{x}_i : \underline{x}_{\mu(i)}) \quad (9.19)$$

QED

The meaning of Claims 14 and 15 is as follows. If $D_{KL}(P \parallel P_T)$ is minimized over all P_T , then

1. P_T inherits its TPMs from P , and
2. P_T gets its structure, which is being parameterized by the function μ , by maximizing the score given by

$$\text{score} = \sum_i H(\underline{x}_i : \underline{x}_{\mu(i)}) . \quad (9.20)$$

(mutual information $H(a : b)$ measures correlation between a and b). Maximizing the score is the same as minimizing the entropy $H(\underline{x}^n)$ over all the structures μ . (i.e., finding least complex structure).

So far, we have studied the properties of those probability distributions P_T for a tree bnet that best approximates an a priori given probability distribution P , but we haven't yet described how to build a Chow-Liu tree based on empirical data. Next we give Chow-Liu's algorithm for doing so.

1. Find MST using Kruskal's algorithm¹. (see Fig.9.1)

Calculate weights $w_{i,j} = H(\underline{x}_i : \underline{x}_j)$ for all $i, j \in V$ and store them in a dictionary D that maps edges to weights.

Order D by weight size.

Let T be a list of the edges in the tree. Initialize T to empty.

Repeat this until T has $n - 1$ elements:

 Remove largest weight w from D and corresponding edge e .

 Add e to T if $\{e\} \cup T$ has no loops. Otherwise discard e and w .

2. Give directions to edges in T . (see Fig.9.2)

Let DT be a list of directed edges. Initialize DT to empty.

Choose any node as root node.

Point arrows along edges in T , away from root node.

Add new arrows to DT .

Repeat this until DT has $n - 1$ elements:

 Point arrows along edges in T , away from leaf nodes of current DT .

 Add new arrows to DT .

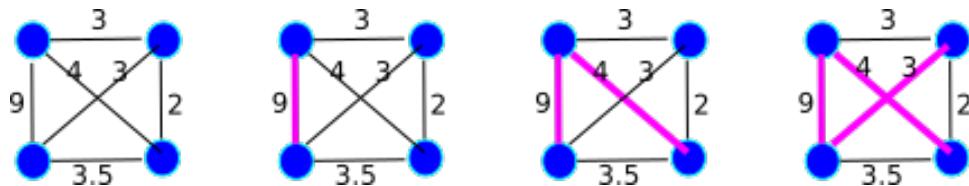


Figure 9.1: Example of finding MST (maximum spanning tree)

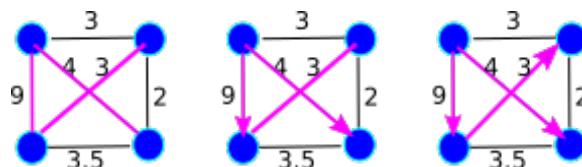


Figure 9.2: Example of giving directions to edges of spanning tree.

Nodes in a Chow-Liu tree can be rated in terms of their relative importance. Here are 2 possible metrics for measuring the importance of a node \underline{a} :

$$N_{nb}(\underline{a}) = \text{ number of neighbors of } \underline{a} \quad (9.21)$$

¹Kruskal's algorithm is one several famous algorithms (Prim's algo is another one) for finding an MST (maximum or minimum spanning tree). An MST algorithm takes an undirected graph with weights along its edges as input. It then finds a tree subgraph (i.e., subset of the edges of the graph with no loops) that (1) spans the graph (i.e., includes every vertex of the graph) and (2) maximizes (or minimizes) the sum of weights among all possible tree subgraphs. For more information, see Ref[110] and references therein, or any other of numerous explanations of MST in the Internet.

$$\text{traffic}(\underline{a}) = \sum_{n \in nb(\underline{a})} H(\underline{a} : n) \quad (9.22)$$

For example, to get a tree with low depth, one can choose as the root node the node which has largest N_{nb} , and if there are several with the same largest N_{nb} , choose out of those the one with the largest traffic.

9.2 Tree Augmented Naive Bayes (TAN)

Recall from Chapter 49 that a Naive Bayes bnet consists of a class node \underline{c} with n children nodes \underline{x}^n , called the feature nodes. A Tree Augmented Naive Bayes (TAN) bnet is a Naive Bayes bnet with a tree grafted onto it like a chimera. More precisely, one starts with a Naive Bayes bnet and adds arrows between the feature nodes. The arrows are added in such a way that the TAN bnet sans node \underline{c} constitutes a tree. It's not the most well motivated bnet in human history, but at least it adds a bit of correlation between the feature nodes of the Naive Bayes bnet. Those nodes are independent at fixed \underline{c} in the Naive Bayes bnet, but are no longer so in the TAN bnet. See Figs.9.3 and 9.4 for an example of a TAN bnet.

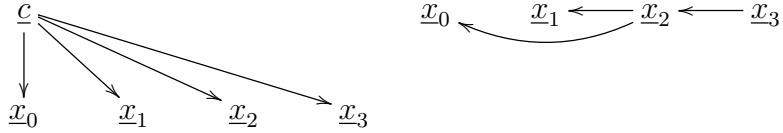


Figure 9.3: bnet for Naive Bayes with 4 feature nodes and another bnet for a tree made of the same feature nodes.

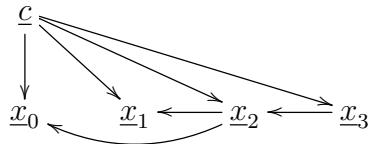


Figure 9.4: TAN bnet constructed by merging Naive Bayes bnet and tree bnet of Fig.9.3.

The total probability distribution P_{TAN} for a TAN bnet can be parameterized as follows.

$$P_{TAN}(x^n, c) = P_{TAN}(c) \prod_{i=0}^{n-1} P_{TAN}(x_i | x_{\mu(i)}, c). \quad (9.23)$$

As with Chow Liu trees, we can attempt to find a TAN bnet whose total probability $P_{TAN} = P_{\underline{x}^n, \underline{c}}$ best approximates an a priori given probability distribution $P = P_{\underline{x}^n, \underline{c}}$.

Note that

Claim 16

$$\operatorname{argmin}_{\mu} H(\underline{x}^n, \underline{c}) = \operatorname{argmax}_{\mu} \sum_i H(\underline{x}_i : \underline{x}_{\mu(i)} | \underline{c}) \quad (9.24)$$

proof:

$$H(\underline{x}^n, \underline{c}) = - \sum_{x^n, c} P(x^n, c) \left[\ln P(c) + \sum_i \ln P(x_i | x_{\mu(i)}, c) \right] \quad (9.25)$$

$$= - \sum_{x^n, c} P(x^n, c) \left[\ln P(c) + \sum_i \ln \left(\frac{P(x_i, x_{\mu(i)} | c)}{P(x_i | c) P(x_{\mu(x_i)} | c)} P(x_i | c) \right) \right] \quad (9.26)$$

$$= \sum_i H(\underline{x}_i, \underline{c}) - \sum_i H(\underline{x}_i : \underline{x}_{\mu(i)} | \underline{c}) \quad (9.27)$$

QED

Following the same line of reasoning that we followed for Chow-Liu trees, we conclude that:

If $D_{KL}(P \| P_{TAN})$ is minimized over all P_{TAN} , then

1. P_{TAN} inherits its TPMs from P , and
2. P_{TAN} gets its structure, which is being parameterized by the function μ , by maximizing the score defined by

$$\text{score} = \sum_i H(\underline{x}_i : \underline{x}_{\mu(i)} | \underline{c}) \quad (9.28)$$

One can build a TAN bnet from empirical data as follows:

Calculate a Chow-Liu Tree for each $c \in S_{\underline{c}}$. For each of those trees, create a TAN bnet, and calculate its score given by Eq.(9.28). Keep the TAN bnet with the largest score.

Chapter 10

Counterfactual Reasoning

10.1 The 3 Rungs of Causal AI

According to Judea Pearl, there are 3 rungs in the ladder of causal AI. These are (as I see them):

1. **Observing Passively. Answering “What next?”:** Collecting data and fitting curves to it, without any plan designed to investigate Nature’s causal connections. Predicting the future.
2. **Doing causal experiments. Answering “Why?”:** Doing experiments consciously designed to elucidate Nature’s causal connections. Even cats do this!, but current AI doesn’t.
3. **Imagining counterfactual situations, Analogizing. Answering “What if?”:** Imagining gedanken experiments to further understand Nature’s causal connections, and to decide what future courses of action are more likely to succeed, even if those courses of action are unprecedented, and have never been taken before. Making predictions about events that have never happened (“counterfactuals”) is a very Bayesian concern, well out of the purview of frequentists. Nevertheless, humans do such “analogizing” all the time to great advantage. It becomes possible if there is some foreign but similar data that can be transported (transplanted, applied) to the situation of interest.

Chapter 43 on message passing is about rung 1. Chapter 17 on Do Calculus is about rung 2. This chapter is dedicated to rung 3.

Judea Pearl is fond of discussing rung 3 solely in terms of SCM.¹ In this chapter, we define rung 3 without using SCM, using solely bnets. This gives a more general version of rung 3, because SCM are a subset of bnets.

¹SCM are what we call DEN. DEN (deterministic systems with external noise) are discussed in Chapter 38.

We will use the term **intervention operator** (or simply “intervention”) to refer to an operator that maps a bnet to another bnet. In Chapter 17, we introduced an intervention operator called the **do operator** $\mathcal{D}_{\underline{x}=x}$ (this is our notation for what Pearl symbolizes by $do(\underline{x}) = x$). The study of counterfactuals requires that we introduce a new kind of intervention operator that we will call an **imagine operator**, and denote by $\mathcal{I}_{\underline{x} \rightarrow \underline{y}}(\tilde{x})$. These 2 types of interventions will be defined next.

10.2 Do operator

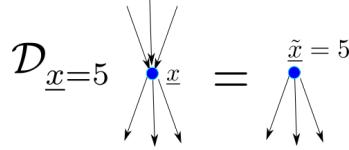


Figure 10.1: Action of “do” operator $\mathcal{D}_{\underline{x}=5}$ on node \underline{x} .

The do operator $\mathcal{D}_{\underline{x}=5}$ is defined graphically in Fig.10.1. The TPM, printed in blue, for node \tilde{x} of Fig.10.1, is as follows.

$$P(\tilde{x}) = \delta(5, \tilde{x}) \quad (10.1)$$

The do operator $\mathcal{D}_{\underline{x}=5}$ amputates the incoming arrows of node \underline{x} and sets the TPM of the new root node \tilde{x} to a delta function $\delta(\tilde{x}, 5)$ (or some state of \underline{x} other than 5). Sometimes we call the new node $\mathcal{D}\underline{x}$ instead of \tilde{x} .

The uses of the do operator are discussed in detail in Chapter 17.

10.3 Imagine operator

The imagine operator $\mathcal{I}_{\underline{x} \rightarrow \underline{y}}(5)$ is defined graphically in Fig.10.2. Note that Fig.10.2 actually defines two types of imagine operators, the one with an argument: $\mathcal{I}_{\underline{x} \rightarrow \underline{y}}(5)$, and the one without an argument: $\mathcal{I}_{\underline{x} \rightarrow \underline{y}}$. The TPMs, printed in blue, for various nodes in Fig.10.2, are as follows.

- For $\mathcal{I}_{\underline{x} \rightarrow \underline{y}}(\tilde{x})G$

$$P(y|\tilde{x}, a.) = P(y|\underline{x} = \tilde{x}, a.) \quad (10.2)$$

$$P(\tilde{x}) = \delta(\tilde{x}, 5) \quad (10.3)$$

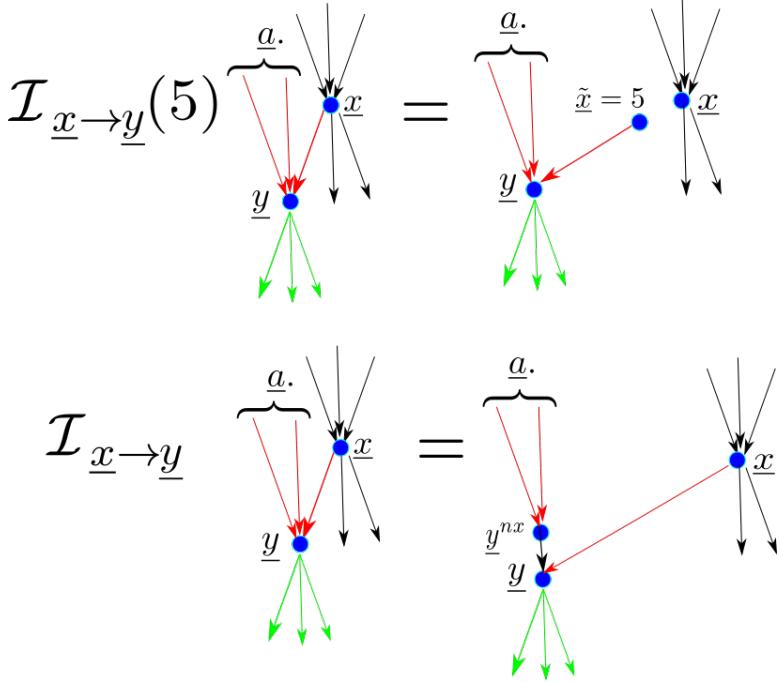


Figure 10.2: Action of “imagine” operators $\mathcal{I}_{\underline{x} \rightarrow \underline{y}}(5)$ and $\mathcal{I}_{\underline{x} \rightarrow \underline{y}}$ on arrow $\underline{x} \rightarrow \underline{y}$. In this figure, $y^{nx} = [y(x)]_{\forall x \in S_{\underline{x}}}$, where $nx = |S_{\underline{x}}|$ and $S_{\underline{x}}$ is the set of states of node \underline{x} .

- For $\mathcal{I}_{\underline{x} \rightarrow \underline{y}} G$

$$P(y^{nx}|a.) = \prod_{\underline{x}} P(\underline{y}(\tilde{\underline{x}}) = y(\tilde{\underline{x}})|a.) \quad (10.4)$$

$$P(y|y^{nx}, x) = \delta(y, y(x)) \quad (10.5)$$

The imagine operators $\mathcal{I}_{\underline{x} \rightarrow \underline{y}}(5)$ and $\mathcal{I}_{\underline{x} \rightarrow \underline{y}}$ operate on an arrow whereas the \mathcal{D} operator operates on a node. $\mathcal{I}_{\underline{x} \rightarrow \underline{y}}(5)$ deletes arrow $\underline{x} \rightarrow \underline{y}$ and creates a new root node $\tilde{\underline{x}}$ and a new arrow $\tilde{\underline{x}} \rightarrow \underline{y}$. Sometimes we call the new node $\mathcal{I}_{\underline{y} \rightarrow \underline{x}}$ instead of $\tilde{\underline{x}}$. $\mathcal{I}_{\underline{x} \rightarrow \underline{y}}$ creates a new node \underline{y}^{nx} and an arrow $\underline{y}^{nx} \rightarrow \underline{y}$.

Fig.10.3 shows how the imagine operator arises in Potential Outcomes (PO) theory. PO theory is discussed extensively in Chapter 56. As you can see, PO theory only uses a limited version of the 3 rungs of causal inference, because it doesn't use the do-operator, and it only uses one of 2 possible types of imagine operators. Furthermore, it assumes a very limited triangular DAG.

Fig.10.4 gives a connection between do and imagine operators. We see from

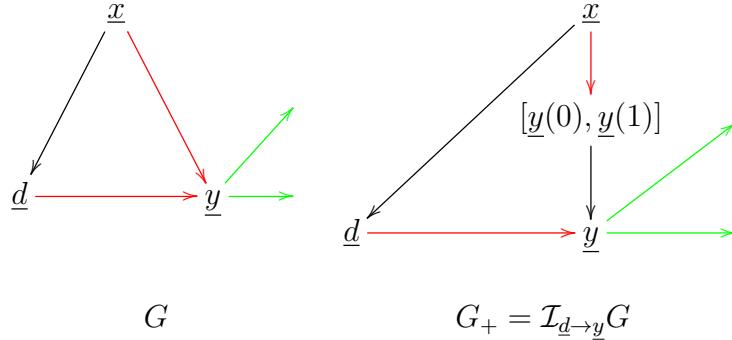


Figure 10.3: How imagine operator arises in Potential Outcomes (PO) theory.

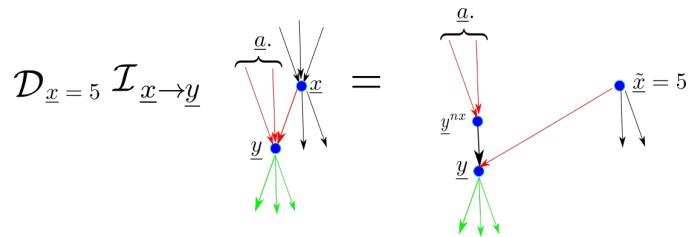


Figure 10.4: $\mathcal{D}_{\underline{x}=5} \mathcal{I}_{\underline{x} \rightarrow \underline{y}} G$ gives a connection between do and imagine operators.

that figure that for $\mathcal{D}_{\underline{x}=\tilde{x}} \mathcal{I}_{\underline{x} \rightarrow \underline{y}} G$, we have²

$$P(y | \mathcal{D}_{\underline{x}} = \tilde{x}, a.) = P(y(\tilde{x}) = y | a.) \quad (10.6)$$

One can define a **do-imagine-calculus** whose objective is to express probabilities such as $P(y | \mathcal{D}_{\underline{L}} = r, \mathcal{I}_{\underline{b}\underline{s}} = s, t)$ in terms of observable probabilities that do not contain any do or imagine operators in them. As with Do Calculus, this reduction is not always possible, and we say a probability is **\mathcal{D} -identifiable**, **\mathcal{I} -identifiable** or **\mathcal{DI} -identifiable** if it can be expressed without do, imagine or both operators.

²In the notation favored by Pearl, Eq.(10.6) would be

$$P(y | do(X) = \tilde{x}, a.) = P(Y_{\tilde{x}} = y | a.)$$

Chapter 11

Cross-Validation

This chapter is based on Ref.[82].

Cross-Validation (CV) is a method of calculating the “**out-of-training-set**” (**OOTS**) error for a classifier. What this means is that the classifier is trained on a training set, and its propensity to err is evaluated on a set different from the training set.

In **k -fold CV**, the most common CV method, and the only one we will discuss in this chapter, one partitions a dataset into k disjoint datasets of equal length. One uses $k - 1$ of those sub-datasets to train a model, and saves the last sub-dataset to validate the model just trained. One actually rotates which of the k sub-datasets is used for validation purposes, and calculates k validation errors \mathcal{E}_j for $j = 0, 1, \dots, k - 1$. Then one averages over the \mathcal{E}_j to obtain a final OOTS error \mathcal{E} .

CV strongly resembles Jackknife Resampling (JR) (see Chapter 34), but in JR the validation sub-dataset is never used for anything, whereas in CV, it is used for validation purposes, to calculate an OOTS error.

Next, we will explain k -fold CV more explicitly, using equations and a bnet.

Let $L = [0, 1, 2, \dots, nsam - 1]$ be a list of individuals (samples) in a population. In this chapter, we will use the notation $A^\sigma = A[\sigma]$ and $\vec{A} = [A^\sigma : \sigma \in L]$ for a list (vector, 1-D array) indexed by L . We will refer to $DS = (\vec{x}, \vec{y})$ where $x^\sigma \in S_x$, $y^\sigma \in S_y$, as a dataset. If L_j is a list (possibly with duplicate items) such that $set(L_j) \subset set(L)$, then define $DS_j = (\vec{x}, \vec{y})_{L_j} = ((x^\sigma)_{\sigma \in L_j}, (y^\sigma)_{\sigma \in L_j})$. We will refer to DS_j as the **restriction of (\vec{x}, \vec{y}) to L_j** .

Let $J = \{0, 1, 2, \dots, nj - 1\}$.

Define a **training list(TL), validation list(VL) pair** (TL, VL) to be a pair of lists such that $set(TL)$ and $set(VL)$ are disjoint subsets of $set(L)$. Let (TL_j, VL_j) for $j \in J$ be nj such TL-VL pairs.

Fig.11.1 shows the TL-VL pairs that are used when doing k -fold CV. In that figure, $k = nj = 4$. As you can see, in k -fold CV, one chooses $nj = k$ list pairs (TL_j, VL_j) such that all individuals $\sigma \in L$ appear exactly once, in either TL_j or VL_j , but not in both.

We will refer to a function $Y : S_x \rightarrow S_c$ as a classifier. It maps a vector of

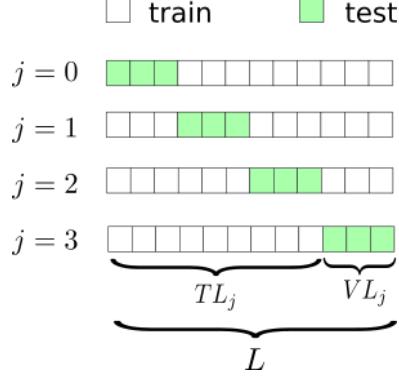


Figure 11.1: 4-fold CV with $|L| = 12$. For all j , $|VL_j| = 3$ and $|TL_j| = 9$. All individuals $\sigma \in L$ appear exactly once, in either TL_j or VL_j , but not in both.

features x to a class c . Let Y_j for $j \in J$ denote n_j classifiers.

If $Y_j : S_{\underline{x}} \rightarrow S_c$ and S_c is a discrete set (“categorical”), then define the **OOTs error for the j th classifier** as:

$$\mathcal{E}_j = \frac{1}{|VL_j|} \sum_{\sigma \in VL_j} \mathbb{1}(y^\sigma \neq Y_j(x^\sigma)) . \quad (11.1a)$$

On the other hand, if $S_c = \mathbb{R}$, it makes more sense to define \mathcal{E}_j as a mean square error:

$$\mathcal{E}_j = \frac{1}{|VL_j|} \sum_{\sigma \in VL_j} (y^\sigma - Y_j(x^\sigma))^2 . \quad (11.1b)$$

Finally, define the **final OOTs error** as

$$\mathcal{E} = \frac{1}{|J|} \sum_{j \in J} \mathcal{E}_j . \quad (11.2)$$

Fig.11.2 gives a bnet that represents the CV algorithm. The TPMs, printed in blue, for the bnet Fig.11.2, are as follows:

$$P((\vec{x}, \vec{y})_{TL_j} | (\vec{x}, \vec{y})) = \mathbb{1}(\text{ } (\vec{x}, \vec{y})_{TL_j} \text{ = restriction of } (\vec{x}, \vec{y}) \text{ to } TL_j.) \quad (11.3)$$

$$P((\vec{x}, \vec{y})_{VL_j} | (\vec{x}, \vec{y})) = \mathbb{1}(\text{ } (\vec{x}, \vec{y})_{VL_j} \text{ = restriction of } (\vec{x}, \vec{y}) \text{ to } VL_j.) \quad (11.4)$$

$$P(Y_j | (\vec{x}, \vec{y})_{TL_j}) = \mathbb{1}(\text{ } Y_j \text{ = classifier trained with } (\vec{x}, \vec{y})_{TL_j} \text{ }) \quad (11.5)$$

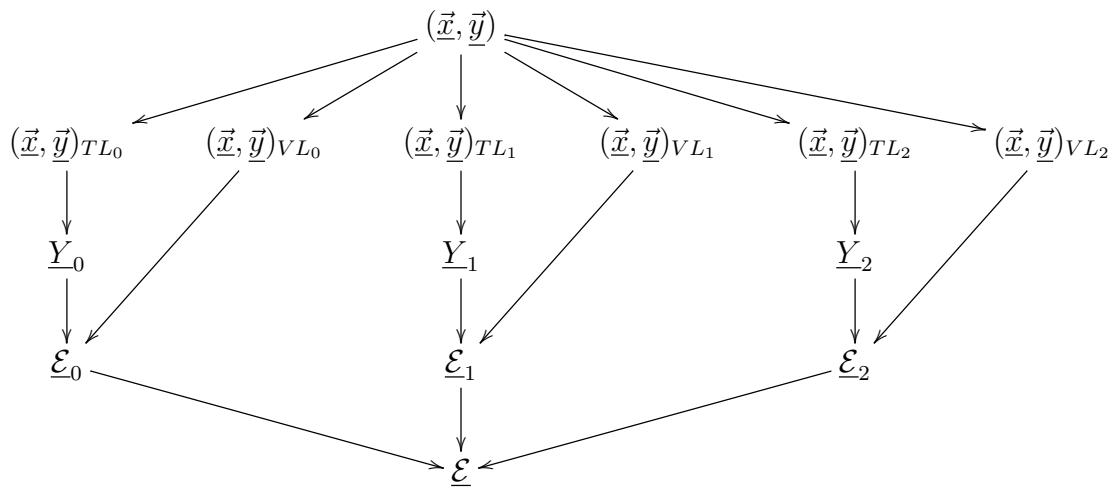


Figure 11.2: Bnet for 3-fold CV.

$$P(\mathcal{E}_j | Y_j, (\vec{x}, \vec{y})_{VL_j}) = \mathbb{1}(\mathcal{E}_j = \text{defined by Eqs.(11.1).}) \quad (11.6)$$

$$P(\mathcal{E} | (\mathcal{E}_j)_{j \in J}) = \mathbb{1}(\mathcal{E} = \text{defined by Eq.(11.2).}) \quad (11.7)$$

Chapter 12

Dataset Shift and Batch Normalization

In this chapter, we will represent Linear Regression (LR) as follows. We list a dataset; i.e., a set of tuples indexed by the individuals σ of a population Σ such that $|\Sigma| = nsam$. The independent variables of the LR (i.e., x^σ) are unboxed and the dependent variable (aka target feature) (i.e., y^σ) is shown inside a box. Then we show an arrow with the superscript “LR-fit”, followed by the fit function obtained by performing the LR.

$$\{(\sigma, x^\sigma = [x_i^\sigma], \boxed{y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{LR-fit}} \hat{y}(x) = \alpha + \sum_i x_i \beta_i \quad (12.1)$$

Analogously, we represent Supervised Machine Learning (ML) as follows.

$$\{(\sigma, x^\sigma, \boxed{y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \hat{y}(x) \quad (12.2)$$

When doing ML, we partition the full population Σ_{full} into two disjoint sets, the **training set** $\Sigma_{train} = \Sigma(\underline{s} = 0) = \Sigma$ and the **testing set** $\Sigma_{test} = \Sigma(\underline{s} = 1) = \Sigma^*$. Then we do two ML fits:

$$\begin{aligned} \text{training: } & \{(\sigma, x^\sigma, \boxed{y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \hat{y}(x) \\ \text{testing: } & \{(\sigma, x^\sigma, \boxed{y^\sigma}) : \sigma \in \Sigma^*\} \xrightarrow{\text{ML-fit}} \hat{y}^*(x) \end{aligned} \quad (12.3)$$

Ideally, $\hat{y}(x)$ and $\hat{y}^*(x)$, will be almost equal for all x . Dataset shift occurs when this is not the case. Equivalently, let

$$\begin{aligned} P_{train}(x, y) &= P(x, y | \underline{s} = 0) = P(x, y) \\ P_{test}(x, y) &= P(x, y | \underline{s} = 1) = P^*(x, y) \end{aligned} \quad (12.4)$$

We say there is a **dataset shift** if

$$P(x, y) \neq P^*(x, y) \quad (12.5)$$

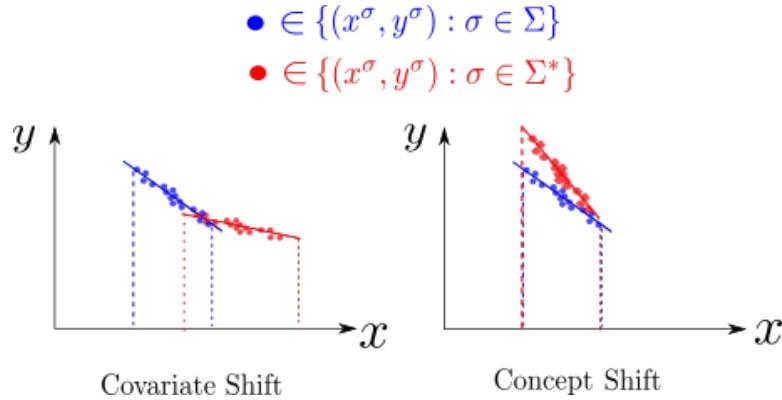


Figure 12.1: For Linear Regression, 2 types of dataset shift: covariate shift and concept shift.

12.1 Covariate Shift

We say there is a **covariate shift** if (see Fig.12.1)

$$P(y|x) = P^*(y|x) \text{ but } P(x) \neq P^*(x) \quad (12.6)$$

This can be represented in terms of bnets as follows¹

$$\begin{array}{ccc}
 \underline{s} = 0 & & \underline{s} = 1 \\
 \downarrow & & \downarrow \\
 \underline{x} \longrightarrow \underline{y} & \neq & \underline{x} \longrightarrow \underline{y} \\
 \underbrace{\hspace{1cm}}_{\underline{s} = 0} & & \\
 = & & \\
 \underline{x} \longrightarrow \underline{y} & &
 \end{array} \quad (12.7)$$

12.2 Concept Shift

We say there is a **concept shift** if (see Fig.12.1)

$$P(y|x) \neq P^*(y|x) \text{ but } P(x) = P^*(x) \quad (12.8)$$

This can be represented in terms of bnets as follows²

¹See See Chapter 66.

²See See Chapter 66.

$$\begin{array}{ccc}
\underline{s} = 0 & & \underline{s} = 1 \\
& \searrow & \searrow \\
& \underline{x} \longrightarrow \underline{y} & \neq \quad \underline{x} \longrightarrow \underline{y} \\
& \underbrace{\hspace{1cm}}_{\underline{s} = 0} & \\
& \uparrow & \\
& \underline{x} \longrightarrow \underline{y} &
\end{array} \tag{12.9}$$

12.3 Batch Normalization

Batch Normalization (BN) is a technique that is used to diminish dataset shift in Neural Nets.

Let $h_i^{\lambda, \sigma}$ be the output, for individual $\sigma \in \Sigma$, of the i th node of layer λ of a Neural Net (NN). Using the notation of Chapter 50,

$$h_i^{\lambda, \sigma} = \mathcal{A}_i^\lambda(z_i^{\lambda, \sigma}) \tag{12.10}$$

where

$$z_i^{\lambda, \sigma} = \sum_j w_{i|j}^\lambda h_j^{\lambda-1, \sigma} + b_i^\lambda \tag{12.11}$$

Activation functions $\mathcal{A}_i^\lambda : \mathbb{R} \rightarrow \mathbb{R}$ for NNs are discussed in Section 50.1. Suppose the population Σ is partitioned into disjoint batches $\Sigma^{(b)}$ for $b = 1, 2, \dots, B$. Let the set of points $\{z_i^{\lambda, \sigma} : \sigma \in \Sigma^{(b)}\}$ have mean $\mu_i^{\lambda(b)}$ and standard deviation $\sigma_i^{\lambda(b)}$. For any $z_i^{\lambda, \sigma}$ with $\sigma \in \Sigma$, define the BN activation function $\mathcal{A}_{i, BN}^\lambda(\cdot)$ by

$$\mathcal{A}_{i, BN}^\lambda(z_i^{\lambda, \sigma}) = \gamma_i^\lambda \left[\frac{z_i^{\lambda, \sigma} - \mu_i^{\lambda(b)}}{\sigma_i^{\lambda(b)}} \right] + \beta_i^\lambda \quad \text{if } \sigma \in \Sigma^{(b)}, \tag{12.12}$$

where the real valued parameters γ_i^λ and β_i^λ are learned during the optimization process. If node \underline{h}_i^λ of the NN has activation function $\mathcal{A}_i^\lambda : \mathbb{R} \rightarrow \mathbb{R}$, defined a new activation function $\mathcal{A}_{i, new}^\lambda : \mathbb{R} \rightarrow \mathbb{R}$ by the composition of functions

$$\mathcal{A}_{i, new}^\lambda = \mathcal{A}_i^\lambda \circ \mathcal{A}_{i, BN}^\lambda \tag{12.13}$$

Hence, the BN activation function is applied after the linear transformation Eq.(12.11), but before the nonlinear transformation \mathcal{A}_i^λ .

Intuition on why BN diminishes dataset shift: We discussed in Section 50.1 how nonlinear activation functions have a range that is smaller than their domain. Presumably, BN helps to make the range of the activation functions even more concentrated.

Chapter 13

Decision Trees

This chapter is based mainly on Ref.[53].

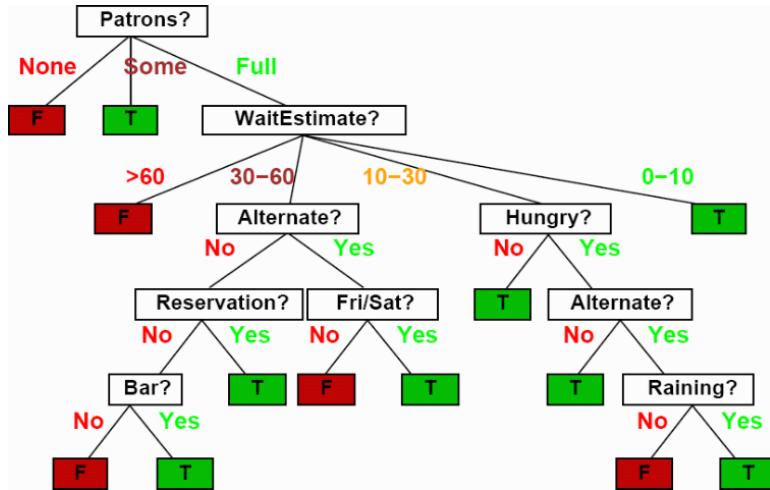


Figure 13.1: Example of dtree taken from Ref.[53]

Fig.13.1 shows a typical decision tree (dtree). This example was taken from Ref.[53], where it is analyzed in detail. As you can see, a dtree contains two main types of nodes: the non-leaf nodes, and the leaf nodes. The **non-leaf nodes** pose **questions**. In general, the **answers**¹ to those questions can be multiple choices with two or more choices. For each of those choices, a tree branch labeled by the choice comes down from the question node. The **leaf nodes** represent endpoints, goals, final conclusions, etc. Dtrees can be viewed as classifiers. They take in a large amount of information about a population and compress that information to just a few classes. If S_c is the set of distinct leaf node labels, then we call each $c \in S_c$ a **class of the classifier**. In the case of Fig.13.1, $S_c = \{False, True\}$.

¹The **question-answer pairs** in dtrees are also called **attribute-value pairs**. Attributes are also called **features**.

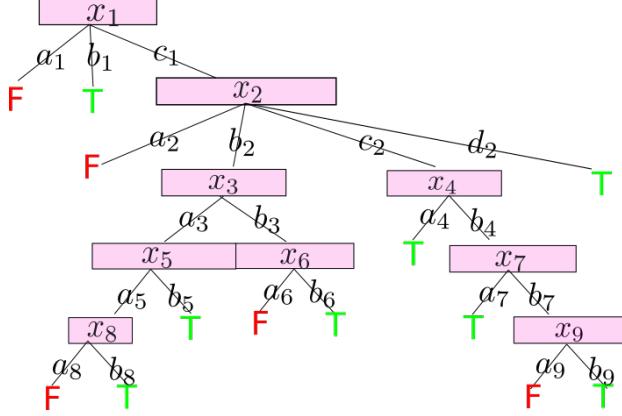


Figure 13.2: Fig.13.1 with abridged labels.

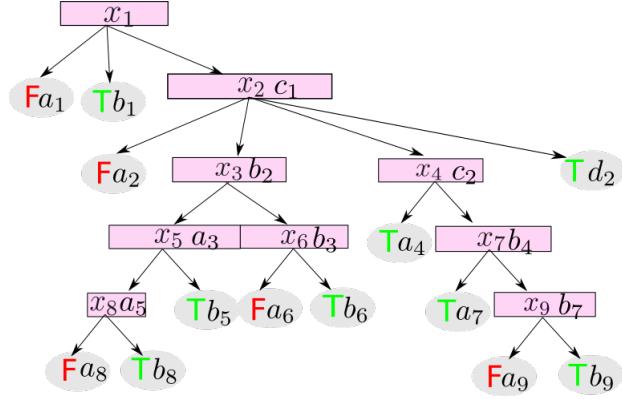


Figure 13.3: Fig.13.2 converted to a bnet.

Dtrees can be used with probabilities attached to each node, or without probabilities (as a plain undirected graph(UG)). This is analogous to bnets, which can be used with probabilities attached to each node (as DAGs with TPMs specified for each node) or without probabilities (as plain DAGs). Dtrees differ from bnets in that their tree branches are labelled, whereas bnet arrows aren't labelled. Also, whereas the nodes of a bnet carry a matrix of probabilities (the TPM), the nodes of a dtree carry just a column vector of probabilities which represents a single probability distribution. Henceforth, we will refer to the column vector of probabilities carried by each node of a dtree as its **Transition Probability Vector (TPV)**. Without the TPVs, a dtree can be used as a deterministic classifier, to classify inputs. With the TPVs, it can be used as a probabilistic sampler (to generate random samples.)

$P(x a)$	$a = a_0$	$a \in S_{\underline{a}}^- - \{a_0\}$	$null$
0	p_0	0	0
1	p_1	0	0
\vdots	\vdots	0	0
$N_{\underline{x}}^- - 1$	$p_{N_{\underline{x}}^- - 1}$	0	0
$null$	0	1	1

Table 13.1: TPM of a node of a dtree image bnet.

13.1 Transforming a dtree into a bnet

A trivial observation that is seldom made in the dtree pedagogical literature is that every dtree maps into a special bnet, let's call it its "image" bnet, in a very natural way. We use the dtree of Fig.13.1 as an example to show how to do this. As a first step, we go from Fig.13.1 to Fig.13.2 by replacing all the labels of the nodes and of the branches of the dtree by abridged symbols. Next, we go from Fig.13.2 to Fig.13.3, by replacing all tree branches by arrows pointing down, and by moving the tree branch labels down so that they become a suffix to the question that the tree branch leads to. At this point, we have created Fig.13.3, which constitutes the DAG of the image bnet. It remains for us to define a TPM for each node of this DAG.

Table 13.1 gives the TPM $P(x|a)$ for a node \underline{x} with single parent \underline{a} of a dtree image bnet. Say node \underline{x} has a set $S_{\underline{x}}^-$ of possible tree branches coming out of it. Let $N_{\underline{x}}^- = |S_{\underline{x}}^-|$. Let $S_{\underline{x}} = S_{\underline{x}}^- \cup \{null\}$ and $N_{\underline{x}} = |S_{\underline{x}}| = N_{\underline{x}}^- + 1$. Define $S_{\underline{a}}^-, N_{\underline{a}}^-, S_{\underline{a}}$ and $N_{\underline{a}}$ analogously for node \underline{a} . In Table 13.1, $S_{\underline{x}}^- = \{0, 1, \dots, N_{\underline{x}}^- - 1\}$ and a_0 is the value of node \underline{a} which labels the tree branch connecting nodes \underline{a} and \underline{x} . $\vec{p} = (p_0, p_1, \dots, p_{N_{\underline{x}}^- - 1})$ is a probability distribution associated with node \underline{x} , its TPV. TPVs can be learned from a dataset following the dtree Structure Learning (SL) algorithm discussed in Section 13.2.

Table 13.1 also applies when node \underline{x} is a leaf node, except that for leaf nodes, \vec{p} is one hot (i.e., all components are zero except for one component which is 1). Also, all leaf nodes \underline{x} have the same $S_{\underline{x}}^-$, namely S_c .

Adding a null state to the set of states (SOS) of each node of the image bnet is necessary because, once $null$ is added to the SOS of any node, it must be added to the SOS of all descendant nodes. $null$ must be added to the SOS of the children of the root node to take care of the situations when those first children don't receive the state they were expecting from their parent, i.e., the root node.

When drawing dtrees, some people put info like explanations and probabilities on the branches of the dtree. That info can all be preserved in the TPM and the node names and node state names of the image bnet nodes. One can also place info inside tool tips attached to the node name and node state names. Often, the pedagogical literature states that dtrees are more explicit and carry more info than their image bnets, but if one follows the above prescriptions, both can carry the same info.

A naive Bayes (NB) bnet (see Chapter 49) consists of a single “class node” with states S_c that fans out with arrows pointing to the “feature nodes”. If each leaf node of a NB bnet fans out into a set of new leaf nodes, and those new leaf nodes also fan out and so on, we get a generalized NB bnet. Let’s call this type of tree bnet an NB^* bnet. An NB^* bnet has the same graph structure as the image bnet of a dtree, but it’s more general, because its TPMs are more general. Each TPM of a NB^* bnet can have several non-trivial columns instead of just one $\text{TPV} = \vec{p}$.

13.2 Structure Learning for Dtrees

Let

$J_0 = \{0, 1, \dots, nj - 1\}$
 $\Sigma = \{0, 1, 2, \dots, nsam - 1\}$
 $DS = \{(\sigma, x^\sigma, c^\sigma) : \sigma \in \Sigma\}$ be a dataset
 $\sigma \in \Sigma$ be an individual (a sample) from a population,
 $x^\sigma \in S_{\underline{x}}$ be the **feature (attributes, questions) vector**. $S_{\underline{x}} = S_{x_0} \times S_{x_1} \times \dots \times S_{x_{nj-1}}$, $x = (x_0, x_1, \dots, x_{nj-1}) \in S_{\underline{x}}$, $x_j \in S_{x_j}$
 $c^\sigma \in S_c$ be a **classification class**
We will assume $S_{\underline{x}}$ and S_c are finite sets.

Building a **classifier** Y (curve fit) for a dtree means finding a deterministic function $Y : S_{\underline{x}} \rightarrow S_c$ such that $c^\sigma \approx Y(x^\sigma)$ for all $\sigma \in \Sigma$. If we divide the population Σ into two large disjoint sets, a **training set** Σ_{train} and a **validation set** Σ_{vali} , and if $c^\sigma \approx Y(x^\sigma)$ very closely for $\sigma \in \Sigma_{train}$ but fits poorly for $\sigma \in \Sigma_{vali}$, then we say the classifier Y suffers from **overfitting**. We can learn the structure and TPVs of a dtree from a dataset DS , by using the dtree **Structure Learning (SL)** algorithm that we will discuss in detail later. However, that algorithm is prone to produce a classifier Y that overfits. Two techniques commonly used to reduce the effects of overfitting are **pruning** and **Random Forest (RF)** (see Chapter 58). Pruning just means somehow removing nodes that are too specific. An RF is an ensemble of dtrees that one averages over. In this chapter, we will only deal with a single dtree, not an ensemble of them.

Dtree SL was invented in 1984-1986 so it is fairly old. Many in the AI community consider dtrees old fashioned compared to neural nets. But dtrees are **interpretable** whereas neural nets aren’t.² Bnets are interpretable too.

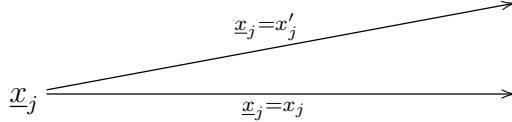
Below, we give the standard algorithm for SL of a dtree, in the form of pseudo-code. But first, we define two quantities, Information Gain and Gini, that are used in that pseudo-code.

²To be precise, only plain dtrees without boosting or bagging are interpretable. Dtrees used within boosting (see Chapter 1 on AdaBoost and Chapter 80 on XGBoost) or bagging (see Chapter 58 on Random Forest) gain much accuracy but lose **interpretability**.

13.2.1 Information Gain, Gini

This section uses various Shannon Information Theory entropies. Our notation for those entropies is described in Chapter Notational Conventions and Preliminaries.

Call a **separation ability measure** (SAM) a measure used to decide, when constructing a dtree from a dataset, in what order to ask the questions about the feature vector x . The question order is decided by searching over all so far unused questions for the question with the largest SAM.³



$$\{N_j(c, x_j)\}_{c \in S_c, x_j \in S_{x_j}}$$

$$\sum_{c \in S_c} N_j(c, x_j) = N_j(x_j)$$

$$\sum_{x_j \in S_{x_j}} N_j(c, x_j) = N_j(c) \quad \sum_{c \in S_c} N_j(c) = \sum_{x_j \in S_{x_j}} N_j(x_j) = N_j$$

Figure 13.4: Some population numbers associated with the nodes of a dtree. $N_j(c, x_j)$ is the number of individuals σ in the population that reaches node j , belonging to class c and having $\underline{x}_j = x_j$.

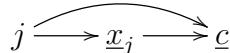


Figure 13.5: Bnet derived from population numbers in Fig.13.4

Fig.13.4 defines some population numbers associated with the nodes of a dtree. From these population numbers, we can define the bnet in Fig.13.5. The TPMs, printed in blue, for the (non-root) nodes of this bnet, are as follows

$$P(c|x_j, j) = \frac{N_j(c, x_j)}{N_j(x_j)} \quad (13.1)$$

$$P(x_j|j) = \frac{N_j(x_j)}{N_j} \quad (13.2)$$

³SAM is also called, somewhat confusingly, the splitting criterion and Gain.

where $j \in J_0$, $x_j \in S_{\underline{x}_j}$, and $c \in S_{\underline{c}}$ is a class node.

One can define the following information theory quantities⁴ associated with the bnet Fig.13.5.

$$INFO_fin_j = - \sum_c P(c|j) \ln P(c|j) \quad (13.3)$$

$$= H(\underline{c}|j) \quad (13.4)$$

$$(13.5)$$

$$INFO_init_j = \sum_{x_j \in S_{\underline{x}_j}} P(x_j|j) H(\underline{c}|x_j, j) \quad (13.6)$$

$$= H(\underline{c}|x_j, j) \quad (13.7)$$

$$IG_j = INFO_fin_j - INFO_init_j \quad (13.8)$$

$$= H(\underline{c}|j) - H(\underline{c}|x_j, j) \quad (13.9)$$

$$= H(\underline{c} : \underline{x}_j | j) \quad (13.10)$$

IG_j is called the **information gain for node \underline{x}_j** . Maximizing this mutual information produces a node \underline{x}_j that has a large correlation to a class c . If the goal is to reach a point where each leaf node is closely correlated to a different class, then maximizing the Information Gain of each new node is a greedy move towards that goal. Thus, Information Gain is a good SAM for dtree SL.

Note that if we approximate

$$\ln P(c|x_j) \approx \ln[1 + P(c|x_j) - 1] \quad (13.11)$$

$$\approx P(c|x_j) - 1 \quad (13.12)$$

in $H(\underline{c}|x_j)$, we get what is called the **Gini (or Gini Index) for node \underline{x}_k** :

$$H(\underline{c}|x_j) = - \sum_c P(c|x_j) \ln P(c|x_j) \quad (13.13)$$

$$\approx 1 - \sum_{c \in S_{\underline{c}}} P(c|x_j)^2 \stackrel{\text{def}}{=} Gini_{x_j} \quad (13.14)$$

⁴The average of $H(\underline{c} : b)$ over b is $H(\underline{c} : \underline{b}) = \sum_b P(b)H(\underline{c} : b)$. Likewise, the average of $H(\underline{c}|b)$ over b is $H(\underline{c}|b) = \sum_b P(b)H(\underline{c}|b)$. $H(\underline{c} : \underline{b})$ becomes $H(\underline{c} : b)$ and $H(\underline{c}|b)$ becomes $H(\underline{c}|b)$ when there is no b -prior (i.e., $P(b)$ is a delta function).

$Gini_{x_j}$ is a fairly good polynomial approximation to $H(\underline{c}|x_j)$. It is computationally much less expensive than $H(\underline{c}|x_j)$, because it does not require computing a log.

We say a probability distribution $P_{\underline{x}}$, is **pure (i.e., deterministic)** if $P_{\underline{x}}(x) = \delta(x, x_0)$. $Gini_{x_j}$ and $H(\underline{c}|x_j)$ are both always non-negative. They both vanish iff $P(c|x_j)$ is pure. Thus, $Gini_{x_j}$ and $H(\underline{c}|x_j)$ are both good measures of **class impurity**.

The **average Gini of node \underline{x}_j** is defined as

$$AGini_j = \sum_{x_j \in S_{\underline{x}_j}} P(x_j|j) Gini_{x_j} . \quad (13.15)$$

It measure the average impurity of the children of node \underline{x}_j .

In practice, the SL algorithm is done recursively. Each recursion step decides which feature x_j will be the root node of the current tree. For all “candidate” features (i.e., all \underline{x}_j that haven’t been used yet as tree nodes), one calculates IG_j , either exactly or approximately via Gini’s, using the following formula:

$$IG_j = \underbrace{H(\underline{c}|j)}_{\approx Gini_j} - \sum_{x_j \in S_{\underline{x}_j}} P(x_j|j) \underbrace{H(\underline{c}|x_j)}_{\approx Gini_{x_j}} \quad (13.16)$$

One then chooses $j = \operatorname{argmax}_j IG_j$. This maximizes the $\underline{c} - \underline{x}_j$ correlation.

Alternatively, some software programs use the average Gini $AGini_j$ as their SAM. They choose as root node $j = \operatorname{argmin}_j AGini_j$. This minimizes the average impurity of the children of node j . Since IG_j differs from $AGini_j$ by $H(\underline{c}|j)$, maximizing IG_j and minimizing $AGini_j$ might lead to different results.

Example of calculation of $AGini_j$ and IG_j

Suppose we deduce from a dataset the numbers in the yellow cells in Table 13.2. These numbers are repeated in Fig.13.6. Then we can calculate the white cells as follows:

$j = temp?$	$x_j = hot$	$x_j = med$	$x_j = cold$
$c = play$	3	3	3
$c = stay$	1	2	2
$Gini$	0.375	0.48	0.48
$AGini = 0.45$			
$H(\underline{c} x_j)$	≈ 0.375	≈ 0.48	≈ 0.48
$IG \approx 0$			

Table 13.2: Evaluating AGini and IG for node *temp*.

- Gini(hot)= $1 - (3/4) - (1/4) = 0.375$

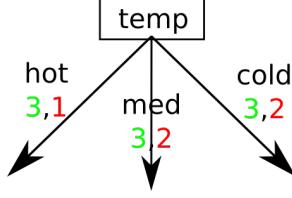


Figure 13.6: Tree stump corresponding to Table 13.2.

- $\text{Gini}(\text{med}) = 1 - (3/5)^2 - (2/5)^2 = 0.48$
- $\text{Gini}(\text{cold}) = 1 - (3/5)^2 - (2/5)^2 = 0.48$
- $\text{AGini} = (4/14)(0.375) + (5/14)(0.48) + (5/14)(0.48) = 0.45$
- $\text{IG} \approx [1 - (6/9)^2 - (3/9)^2] - \text{AGini} = 0.44 - 0.45 \approx 0$

This gives the AGini and IG for just one candidate root node. We would have to calculate AGini (or IG) for all possible candidates and choose the candidate with the lowest AGini (or highest IG).

13.3 Information Gain Ratio

The Information Gain Ratio (IGR) is an alternative SAM.

$$IGR_j = \frac{IG_j}{H(\underline{x}_j|j)} \quad (13.17)$$

$$= \frac{H(\underline{c} : \underline{x}_j|j)}{H(\underline{x}_j|j)} \quad (13.18)$$

$$= \frac{H(\underline{x}_j|j) - H(\underline{x}_j|\underline{c}, j)}{H(\underline{x}_j|j)} \quad (13.19)$$

$$= 1 - \frac{H(\underline{x}_j|\underline{c}, j)}{H(\underline{x}_j|j)} \quad (13.20)$$

$$0 \leq IGR_j \leq 1$$

$$IGR_j = 0 \text{ iff } H(\underline{x}_j|\underline{c}, j) = H(\underline{x}_j|j).$$

13.3.1 Pseudo-code

Below, we give the standard algorithm for SL of a dtree, in the form of pseudo-code. The strategy employed by the algo is to assume an incoming population into the current root node, then determine the feature x_j that best separates that incoming population. The feature x_j is chosen so as to maximize IG_j (or minimize $AGini_j$).

This process is repeated by nominating the end of each new branch to be the current root node. Features can appear as a node more than once, so the order in which nodes are split does not matter. In essence, what we are doing is performing a top-down, greedy search through the space of possible dtrees.

The pseudo-code below describes the following historically important software programs:

- CART (Classification and Regression Trees), invented by Breiman et al in 1984. Uses $AGini_j$ as SAM.
- ID3 (Iterative Dichotomiser 3) invented by Quinlan in 1986. Uses IG_j as SAM. C4.5/C5.0 are successors to ID3.

Thus, CART and ID were invented independently around the same time. The main difference between them is the SAM being used.

The pseudo-code below uses the majority function defined in Chapter Notational Conventions and Preliminaries.

Algorithm 1: Pseudo-code for learning a dtree from a dataset

Input :

- dataset $DS = \{(\sigma, x^\sigma, c^\sigma) : \sigma \in \Sigma\}$
- set of currently available node indices J , where $J \subset J_0$

Output:

- tree T ,
- population numbers $\{(r, c, x_r, N_r(c, x_r)) : r \in J_0, c \in S_c, x_r \in S_{x_r}\}$ stored globally

From DS , calculate J_0, S_c, S_{x_i} for each i
 c^σ is called the target feature/attribute.

$$J \leftarrow J_0$$

Function `learn_dtreet(DS, J):`

```

 $\Sigma \leftarrow$  set of all  $\sigma$  in  $DS$ 
if  $\{c^\sigma : \sigma \in \Sigma\} = \{c\}$  then
     $T \leftarrow$  one node tree with leaf node label=  $c$ 
else if  $J = \emptyset$  then
     $T \leftarrow$  one node tree with leaf node label= majority( $[c^\sigma : \sigma \in \Sigma]$ )
else
     $r \leftarrow \underset{j \in J}{\operatorname{argmax}} IG_j(DS)$  // or replace  $\operatorname{argmax}_{j \in J} IG_j$  by  $\operatorname{argmin}_{j \in J} AGini_j$ 
    from  $DS$ , calculate  $\{(r, c, x_r, N_r(c, x_r)) : c \in S_c, x_r \in S_{x_r}\}$  and store it
        globally
    for  $v \in S_{x_r}$  do
        /* Notice that  $J$  is the same every time repeat this loop,
           so order in which  $v \in S_{x_r}$  are called does not matter.
           Furthermore, this means that multiple tree nodes may be
           labeled by same feature. */
        On current tree  $T$ , add a branch below  $x_r$  with label " $x_r = v$ "
         $DS|_{x_r=v} \leftarrow$  subset of  $DS$  with  $x_r = v$ 
        if  $DS|_{x_r=v} = \emptyset$  then
            below the new branch add a
            leaf node labeled = majority( $[c^\sigma : \sigma \in \Sigma]$ )
        else
            below the new branch add
            subtree = learn_dtreet( $DS|_{x_r=v}, J - \{r\}$ )
    return  $T$ 

```

Chapter 14

**Decisions Based on Rungs 2 and 3:
COMING SOON**

Chapter 15

Difference-in-Differences

This chapter is based on Ref.[6].

The Difference-in-Differences (DID) method was first used by John Snow in an 1854 report that argued that cholera in London was being transmitted by sewage polluted water rather than, as others at the time believed, by air (in fetid vapors called miasmas). In general, one can apply DID to discover causal effects in historical data. By **historical data** (aka a **natural experiment**. See Ref.[116]) we mean data that is collected long after the treatment (rather than during it) and is thus not subject to active intervention by the experimenter.

This chapter assumes that the reader has read Chapter 56 on Potential Outcomes (PO). The DID method applies the basic single-time PO theory described in Chapter 56, to 2 well separated times in which different conditions prevail.

15.1 John Snow, DID and a cholera transmission pathway

Let

$$\begin{aligned} d &\in \{0, 1\} \\ t &\in \{t_0, t_1\}, t_0 < t_1 \\ y &= f(d, t) \in \mathbb{R}. \end{aligned}$$

Define

$$\Delta_t f(d, t) = f(d, t_1) - f(d, t_0), \quad (15.1)$$

$$\Delta_d f(d, t) = f(1, t) - f(0, t), \quad (15.2)$$

$$DID = \delta = \Delta_d \Delta_t f(d, t). \quad (15.3)$$

DID is illustrated in Fig.15.1.

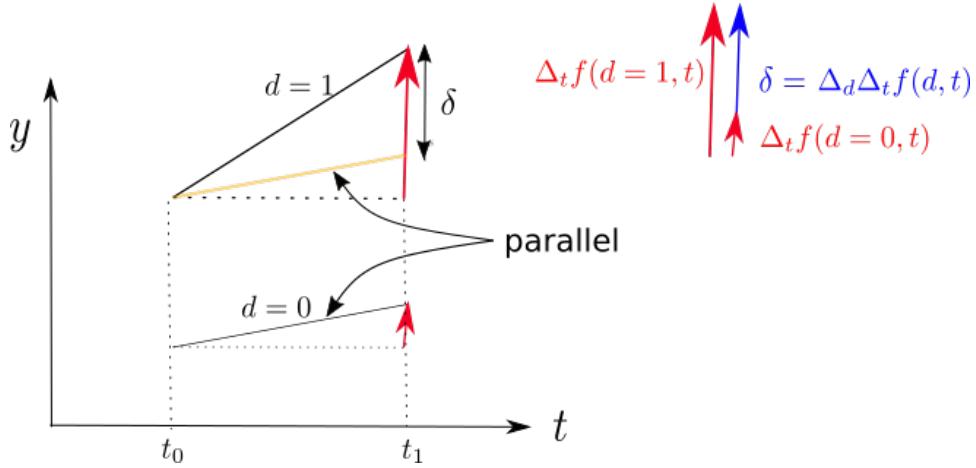


Figure 15.1: Pictorial representation of Difference-in-differences (DID) as a difference of two differences (i.e., a difference of two slopes).

A **time series** is any function of time for which the domain is a discrete set of times.

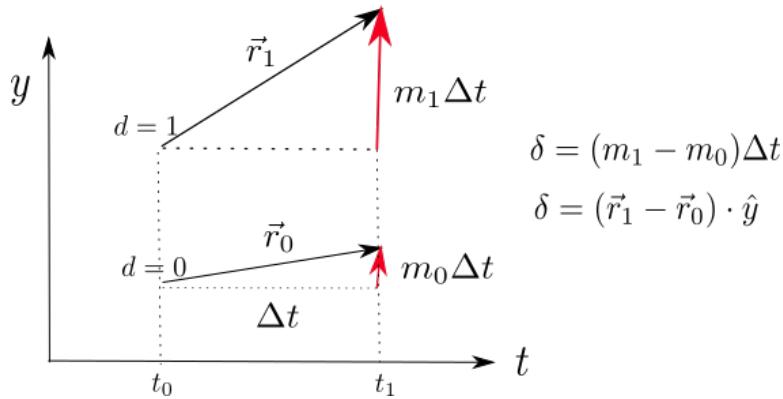


Figure 15.2: $DID = \delta$ expressed as difference of slopes or difference of vectors.

Note that, as shown Fig.15.2, $DID = \delta$ can also be expressed as a difference of 2 slopes times $\Delta t = t_1 - t_0$. Let \hat{y} be a unit in the y direction. δ can also be expressed as the dot product of a difference of 2 vectors dotted with \hat{y} .

A condensation of the data collected by John Snow in 1854 is given in Table 15.1. From that data, we find that

$$\delta = \Delta_d \Delta_t f(d, t) = (19 - 85) - (147 - 135) = -66 - 12 = -78 \quad (15.4)$$

	$t = t_0$ (1849)	$t = t_1$ (1854)
$d = 1$ (town 1)	85 deaths, polluted DW	19 deaths, unpolluted DW
$d = 0$ (town 0)	135 deaths, polluted DW	147 deaths, polluted DW

Table 15.1: A condensation of the data collected by John Snow in 1854, to test the hypothesis that cholera in London was being spread by polluted drinking water (DW).

15.2 PO analysis

In this section, we show how to analyze the DID method using the formalism of PO theory.

We will speak of a treatment outcome $\underline{y}_{t,g^\sigma}^\sigma(c^\sigma, x^\sigma)$ for individual σ that depends, not just on the treatment dose $c^\sigma \in \{0, 1\}$ and the confounder state x^σ , but also on a group parameter (i.e., which population or town) $g^\sigma \in \{0, 1\}$ and on a time parameter $t \in \{t_0, t_1\}$ (note t is independent of σ). Actually, we will assume $g^\sigma = c^\sigma$, so we will just speak of $\underline{y}_t^\sigma(c^\sigma, x^\sigma)$ with no explicit g^σ dependence. As usual for PO theory, we will consider expected values of y_t^σ :

$$E_{\sigma|d,x}[\underline{y}_t^\sigma(c)] = E_{\underline{y}_t(c)|d,x}[\underline{y}_t(c)] = \mathcal{Y}_{c|d,x}(t) \quad (15.5)$$

To calculate these expected values, we need a “model” with probability distributions. In this case, the needed model and probability distributions are provided by the bnets depicted in Fig.15.3. The TPMs, printed in blue, for the bnet $G_{t,+}$ in Fig.15.3, are as follows. Note that the TPMs for the bnet $G_{t,+}$ are defined in terms of the TPMs for the bnet G_t .

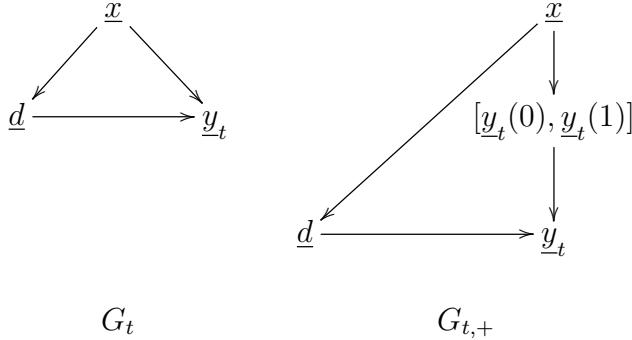


Figure 15.3: $t \in \{t_0, t_1\}$. Bnet $G_{t,+}$ is obtained by adding two new nodes $\underline{y}_t(0)$ and $\underline{y}_t(1)$ to bnet G_t .

$$P(x) = P_{\underline{x}}(x) \quad (15.6)$$

$$P(d|x) = P_{d|\underline{x}}(d|x) \quad (15.7)$$

$$P(y_t|y_t(0), y_t(1), d) = \mathbb{1}(y_t = y_t(d)) \quad (15.8)$$

$$P(y_t(c)|x) = P(y_t(c)|d, x) = \text{given} \quad (15.9)$$

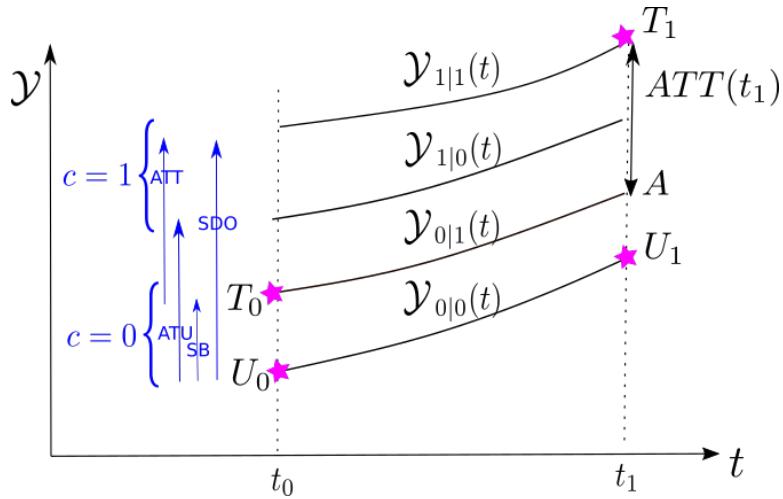


Figure 15.4: Four different time-dependent expected values $\mathcal{Y}_{c|d}(t)$ of y_t^σ for bnet $G_{t,+}$. The 4 magenta stars represent the 4 DID measurements.

We define the function $c(t)$ for $t = t_0, t_1$ by

$$c(t) = \begin{cases} 0 & \text{if } t = t_0 \\ d & \text{if } t = t_1 \end{cases} \quad (15.10)$$

Now we claim that the DID δ calculated in the previous section for John Snow's data, can be expressed in PO formalism as follows:

$$\delta = \Delta_d \Delta_t \mathcal{Y}_{c(t)|d}(t) . \quad (15.11)$$

Fig.15.4 depicts the four functions $\mathcal{Y}_{c|d}(t)$ for t in the interval $[t_0, t_1]$ and for $c, d \in \{0, 1\}$. The \mathcal{Y} coordinates of the four magenta stars in Fig.15.4 can be calculated using bnet G_t .

Define the **parallel trends** (PT) by

$$PT = \Delta_d \Delta_t \mathcal{Y}_{0|d}(t) . \quad (15.12)$$

We will say the **parallel trends assumption (PTA)** holds if $PT = 0$.

Next we prove that the DID δ equals the sum of an ATT¹ and PT.

$$\delta = \Delta_d \Delta_t \mathcal{Y}_{c(t)|d}(t) \quad (15.13)$$

$$= [\Delta_t \mathcal{Y}_{c(t)|1}(t) - \Delta_t \mathcal{Y}_{c(t)|0}(t)] \quad (15.14)$$

$$= \mathcal{Y}_{1|1}(t_1) - \mathcal{Y}_{0|1}(t_0) - \{\mathcal{Y}_{0|0}(t_1) - \mathcal{Y}_{0|0}(t_0)\} \quad (15.15)$$

$$= \mathcal{Y}_{1|1}(t_1) - \mathcal{Y}_{0|1}(t_0) - \{\mathcal{Y}_{0|0}(t_1) - \mathcal{Y}_{0|0}(t_0)\} + \underbrace{\{\mathcal{Y}_{0|1}(t_1) - \mathcal{Y}_{0|1}(t_1)\}}_{\text{zero}} \quad (15.16)$$

$$= \underbrace{\mathcal{Y}_{1|1}(t_1) - \mathcal{Y}_{0|1}(t_1)}_{ATT(t_1)} - \mathcal{Y}_{0|1}(t_0) - \{\mathcal{Y}_{0|0}(t_1) - \mathcal{Y}_{0|0}(t_0)\} + \mathcal{Y}_{0|1}(t_1) \quad (15.17)$$

$$= ATT(t_1) - \Delta_t \mathcal{Y}_{0|0}(t) + \Delta_t \mathcal{Y}_{0|1}(t) \quad (15.18)$$

$$= ATT(t_1) + \underbrace{\Delta_d \Delta_t \mathcal{Y}_{0|d}(t)}_{\text{zero if PTA holds}} \quad (15.19)$$

15.3 Linear Regression

In this section, we show how to apply linear regression (LR) to the PO analysis of DID.

As before, let $y_t^\sigma(c^\sigma)$ be the treatment outcome for individual σ , who receives a treatment dose c^σ at times $t \in \{t_0, t_1\}$. $y_t^\sigma(c^\sigma)$ can be fitted as follows. Here ϵ^σ is the residual for individual σ , and $b_0, m_0, b_1, m_1 \in \mathbb{R}$ are the fit parameters.

$$y_t^\sigma = [b_0 + m_0(t - t_0)](1 - c^\sigma) + [b_1 + m_1(t - t_0)]c^\sigma + \epsilon^\sigma. \quad (15.20)$$

Note that Eq.(15.20) yields a straight line in the $y_t^\sigma - t$ plane for $c^\sigma = 0$, and another straight line for $c^\sigma = 1$. We are using the standard symbols b to denote the y-intercept, and m to denote the slope of a straight line.

Taking the expected value of Eq.(15.20), we get

$$\mathcal{Y}_{c|d}(t) = [b_0 + m_0(t - t_0)](1 - d) + [b_1 + m_1(t - t_0)]d. \quad (15.21)$$

If $\Delta t = t_1 - t_0$, then

$$\mathcal{Y}_{d|d}(t_1) = [b_0 + m_0 \Delta t](1 - d) + [b_1 + m_1 \Delta t]d, \quad (15.22)$$

and

$$\mathcal{Y}_{0|d}(t_0) = b_0. \quad (15.23)$$

¹ATT stands for the average treatment effect of the treated. ATT is defined in Chapter 56

Thus,

$$\delta = \Delta_d \Delta_t \mathcal{Y}_{c(t)|d}(t) \quad (15.24)$$

$$= \Delta_d [\mathcal{Y}_{d|d}(t_1) - \mathcal{Y}_{0|d}(t_0)] \quad (15.25)$$

$$= (m_1 - m_0) \Delta t . \quad (15.26)$$

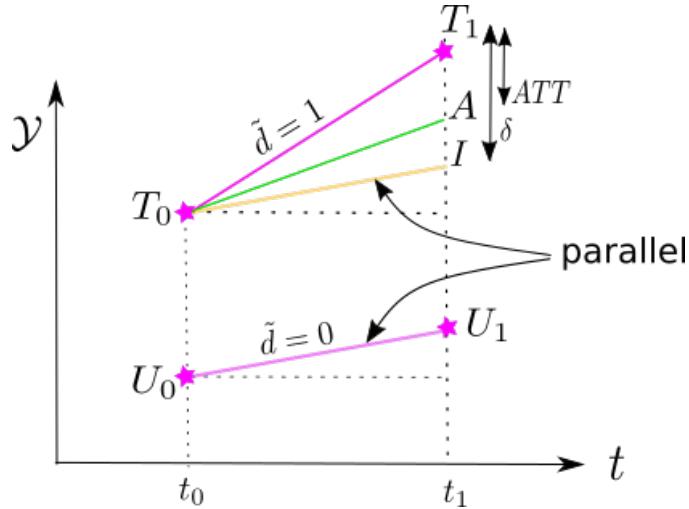


Figure 15.5: We use Linear Regression to fit a straight line between points U_0 and U_1 , and between points T_0 and T_1 . (U =untreated, T =treated, subscript refers to times t_0, t_1). U_0, T_0, U_1, T_1 are the measurement points. Point I is an image of point U_1 .

	$t = t_0$	$t = t_1$
$d = 1$	$\mathcal{Y}(T_0) = \mathcal{Y}_{0 1}(t_0)$	$\mathcal{Y}(T_1) = \mathcal{Y}_{1 1}(t_1)$
$d = 0$	$\mathcal{Y}(U_0) = \mathcal{Y}_{0 0}(t_0)$	$\mathcal{Y}(U_1) = \mathcal{Y}_{0 0}(t_1)$

Table 15.2: \mathcal{Y} coordinates of points U_0, T_0, U_1, T_1 in Figs.15.4 and 15.5.

Figs.15.4 and 15.5 define points U_0, T_0, U_1, T_1, I, A . The \mathcal{Y} coordinates of points U_0, T_0, U_1, T_1 are given by Table 15.2. The \mathcal{Y} coordinates of points A, I are given by Eqs.15.27.

$$\mathcal{Y}(A) = \mathcal{Y}_{0|1}(t_1) \quad (15.27a)$$

$$\mathcal{Y}(I) = \mathcal{Y}(U_1) + [\mathcal{Y}(T_0) - \mathcal{Y}(U_0)] \quad (15.27b)$$

We can express ATT and the δ for DID in terms of the \mathcal{Y} of the points U_0, T_0, U_1, T_1, I, A . Indeed,

$$\delta = \mathcal{Y}(T_1) - \mathcal{Y}(I) \quad (15.28)$$

$$= \mathcal{Y}(T_1) - \mathcal{Y}(U_1) - [\mathcal{Y}(T_0) - \mathcal{Y}(U_0)] \quad (15.29)$$

$$ATT = \mathcal{Y}(T_1) - \mathcal{Y}(A) \quad (15.30)$$

Hence,

$$\delta = ATT \iff \mathcal{Y}(I) = \mathcal{Y}(A) \iff \text{PTA holds} \quad (15.31)$$

Chapter 16

Digital Circuits

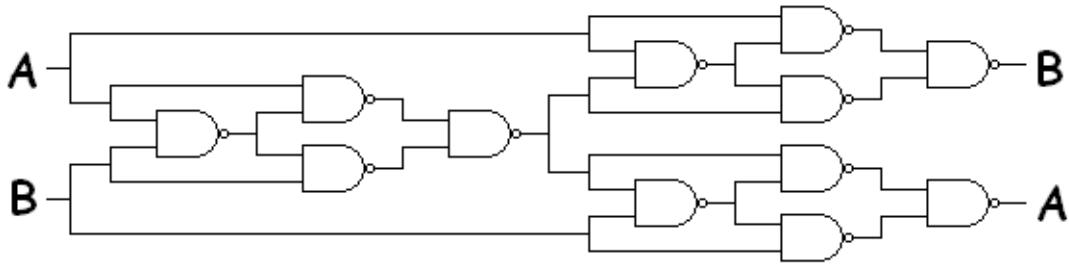


Figure 16.1: Typical digital circuit of NAND gates.

Digital (logic) gate: node with na input ports and nx output ports which represents a function

$$f : \{0, 1\}^{na} \rightarrow \{0, 1\}^{nx}. \quad (16.1)$$

Suppose

$$a^{na} = (a_i)_{i=0,1,\dots,na-1} \text{ where } a_i \in \{0, 1\},$$

$$x^{nx} = (x_i)_{i=0,1,\dots,nx-1} \text{ where } x_i \in \{0, 1\}.$$

f maps a^{na} into x^{nx} .

Digital circuit (dcircuit) = circuit of digital gates.

16.1 Mapping any dcircuit to a bnet

16.1.1 Option A of Fig.16.2

1. Replace every dcircuit gate described by Eq.(16.1) by nx bnet nodes \underline{x}_i for $i = 0, 1, \dots, nx - 1$ such that

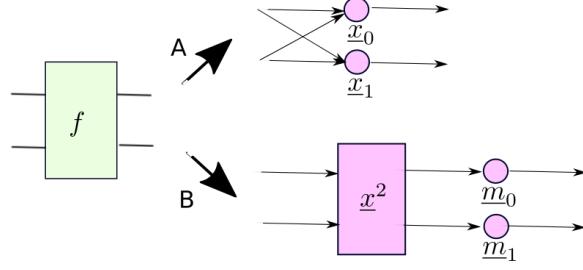


Figure 16.2: 2 options for mapping dcircuit node with multiple output ports into bnet.

$$P(x_i|a^{na}) = \delta(x_i, f_i(a^{na})) \quad (16.2)$$

2. Replace all connectors of the dcircuit by arrows pointing in the direction of the bit flow.

16.1.2 Option B of Fig.16.2

1. Replace every dcircuit gate described by Eq.(16.1) with one bnet node called x^{nx} and, if $nx > 0$, nx “marginalizer nodes” \underline{m}_i for $i = 0, 1, \dots, nx - 1$, such that

$$P(x^{nx}|a^{na}) = \delta(x^{nx}, f(a^{na})) , \quad (16.3)$$

and

$$P(m_i|x^{nx}) = \delta(m_i, x_i) . \quad (16.4)$$

2. Replace all connectors of the dcircuit by arrows pointing in the direction of the bit flow.

Options A and B don't work for digital circuits with feedback loops such as flip-flops. Those could probably be modeled with dynamical bnets.

Chapter 17

Do Calculus

The Do Calculus and associated ideas were invented by Judea Pearl and collaborators. This chapter is based on Judea Pearl's books (see Navigating the ocean of Judea Pearl's Books).

When doing Do Calculus, it is convenient to separate the nodes of a bnet into 2 types: **observed**, and **hidden** (i.e., **unobserved**, **latent**, **unmeasured**, **non-visible**), depending on whether data describing the state of that node is available (i.e., measured) or not. In this chapter, every hidden node will be indicated in a bnet diagram by either: (1) enclosing its random variable in a dashed circle or (2) making the arrows coming out of it dashed. Accordingly, the 3 diagrams in Fig.17.1 all mean the same thing.

A **confounder node** \underline{c} for nodes \underline{x} and \underline{y} is a root node with arrows pointing from it to both \underline{x} and \underline{y} . Thus, \underline{c} acts as a **common cause** of \underline{x} and \underline{y} . In general, confounders can be either observed or hidden nodes. The word “confounder” itself just means that it confuses the analysis. It says nothing about whether it is hidden or not. The node \underline{c} in Fig.17.1) is a **hidden confounder**.

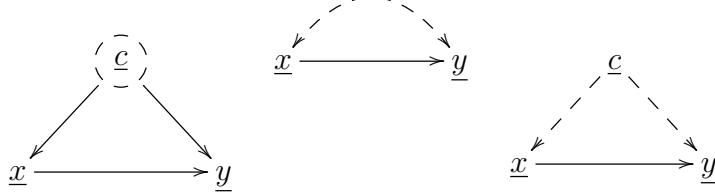


Figure 17.1: These 3 diagrams are equivalent. They mean that node \underline{c} is hidden. Node \underline{c} is implicit in the middle diagram.

Let $\mathcal{D}_{\underline{x}}$ be an operator that acts on a graph G with a node \underline{x} by deleting all the arrows entering \underline{x} , thus converting \underline{x} into a new node $\mathcal{D}\underline{x}$ that is a root node. Let $\mathcal{L}_{\underline{x}}$ be an operator that acts on a graph G with a node \underline{x} by deleting all the arrows leaving \underline{x} , thus converting \underline{x} into a new node $\mathcal{L}\underline{x}$ that is a leaf node. $\mathcal{D}_{\underline{x}}$ and $\mathcal{L}_{\underline{x}}$ are depicted in Fig.17.2. ¹

¹Pearl uses $\mathcal{D}_X G = G_{\overline{X}}$ and $\mathcal{L}_X G = G_{\underline{X}}$ for a random variable X in a graph G . The way I

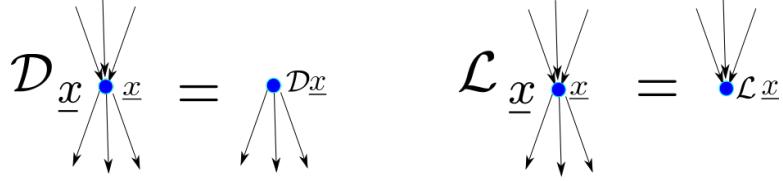


Figure 17.2: The do operator $\mathcal{D}_{\underline{x}}$ converts node \underline{x} into a root node $\mathcal{D}\underline{x}$. The leaf operator $\mathcal{L}_{\underline{x}}$ converts node \underline{x} into a leaf node $\mathcal{L}\underline{x}$.

If you don't know yet what we mean by a multi-node $\underline{a}.$, see Chapter Definition of a Bayesian Network

Given a bnet G , we define as follows the operators $\mathcal{D}_{\underline{a}.$ and $\mathcal{L}_{\underline{a}.$ for a multi-node $\underline{a}..$

$$\mathcal{D}_{\underline{a}.}G = \left[\prod_j \mathcal{D}_{\underline{a}_j} \right] G, \quad \mathcal{L}_{\underline{a}.}G = \left[\prod_j \mathcal{L}_{\underline{a}_j} \right] G. \quad (17.1)$$

Consider a bnet whose totality of nodes is labeled $\underline{X}..$ Recall that

$$P(X.) = \prod_j P(X_j | (X_k)_{k: \underline{X}_k \in pa(\underline{X}_j)}). \quad (17.2)$$

Define an operator \mathcal{D} that acts as follows²: Let $X. - a. = (X_k)_{k: \underline{X}_k \notin \underline{a}.}$

$$P(X. - a. | \mathcal{D}\underline{a}. = a.) = \mathcal{N}(!(\underline{X} - a.)) \frac{P(X.)}{\prod_{j: \underline{X}_j \in \underline{a}.} P(X_j | (X_k)_{k: \underline{X}_k \in pa(\underline{X}_j)})} \quad (17.3)$$

$$= \mathcal{N}(!(\underline{X} - a.)) \prod_{j: \underline{X}_j \notin \underline{a}.} P(X_j | (X_k)_{k: \underline{X}_k \in pa(\underline{X}_j)}) \quad (17.4)$$

$$\neq P(X. - a. | \underline{a}. = a.). \quad (17.5)$$

Also,

$$P(\mathcal{D}\underline{a}. = a.) = \delta(a'., a.). \quad (17.6)$$

In words, we replace the TPM for multinode $\underline{a}.$ by a delta function.

For instance, for the bnet

$$\underline{r} \longrightarrow \underline{x} \longrightarrow \underline{y} \quad (17.7)$$

with

remember Pearl's notation is top-in (as in topping), and bottom-out (as in butt-out).

²As usual, $\mathcal{N}(!x)$ denotes a constant that is independent of x .

$$P(r, x, y) = P(y|x)P(x|r)P(r), \quad (17.8)$$

one has

$$P(r, y|\mathcal{D}\underline{x} = x) = P(y|x)P(r) \quad (17.9)$$

Hence,

$$P(y|\mathcal{D}\underline{x} = x) = P(y|x) \quad (17.10)$$

For the bnet



with

$$P(x, y, c) = P(y|x, c)P(x|c)P(c), \quad (17.12)$$

one has

$$P(y, c|\mathcal{D}\underline{x} = x) = P(y|x, c)P(c). \quad (17.13)$$

Hence,

$$P(y|\mathcal{D}\underline{x} = x) = \sum_c P(y|x, c)P(c). \quad (17.14)$$

This is called **adjusting the parents of \underline{x}** .

For $\underline{b} \subset \underline{X} - \underline{a}$, define

$$P(b|\mathcal{D}\underline{a} = a) = \sum_{X.-a.-b} P(X.-a|\mathcal{D}\underline{a} = a), \quad (17.15)$$

and for $\underline{s} \subset \underline{X} - \underline{a} - \underline{b}$, define

$$P(b, s|\mathcal{D}\underline{a} = a, s) = \frac{P(b, s|\mathcal{D}\underline{a} = a)}{P(s|\mathcal{D}\underline{a} = a)}. \quad (17.16)$$

$P(b|\mathcal{D}\underline{a} = a, s)$ is denoted by Pearl by $P(b|do(\underline{a} = a), s)$. I prefer to use \mathcal{D} instead of $do()$. I will still call \mathcal{D} a **do operator**.

In $P(y|\mathcal{D}\underline{x} = x)$, node \underline{x} is turned into a root node. This guarantees that there is no confounding node connecting \underline{x} and \underline{y} . Such confounding nodes are unwelcomed when calculating causal effects between the 2 variables \underline{x} and \underline{y} because they introduce non-causal correlations between the two. This is also what happens in a **Randomized Controlled Trial (RCT)**. In an RCT with treatment \underline{x} , the value of \underline{x} for each

patient is determined by a coin toss, effectively turning \underline{x} into a root node. Hence, the do operator mimics an RCT.

$P(b.|D\underline{a}_. = a., s.)$ is said to be **do-identifiable** (i.e., expressible without do()) if it can be expressed in terms of probability distributions that only depend on observed variables, and that have no do operators in them.³

For $\underline{x}, \underline{y} \in \{0, 1\}$, the **average controlled effect (ACE)** is defined as

$$ACE = P(y = 1|D\underline{x} = 1) - P(y = 1|D\underline{x} = 0) \quad (17.17)$$

and the **Risk Difference (RD)** is defined as

$$RD = P(y = 1|\underline{x} = 1) - P(y = 1|\underline{x} = 0). \quad (17.18)$$

17.1 3 Rules of Do Calculus

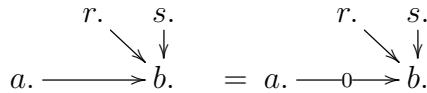
Throughout this section, suppose $\underline{a}_., \underline{b}_., \underline{r}_., \underline{s}_.$ are disjoint multinodes in a bnet G .

Recall from Chapter 19 on d-separation, that $(\underline{b}_. \perp_G \underline{a}_. | \underline{r}_., \underline{s}_.)$ means that we have established from the d-separation rules that all paths in G from $\underline{a}_.$ to $\underline{b}_.$ are blocked if we condition on $\underline{r}_. \cup \underline{s}_..$ Recall also that:

Rule 0: Insertion or deletion of observations, without do operators.

If $(\underline{b}_. \perp \underline{a}_. | \underline{r}_., \underline{s}_.)$ in G , then

✓ $P(b.|a., r., s.) = P(b.|r., s.) \quad (\text{i.e., } \underline{a}_. = a. \leftrightarrow 1)$



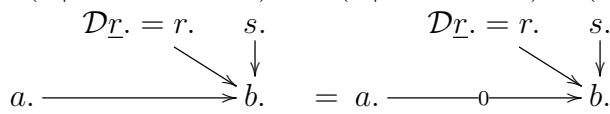
✓ $H(\underline{b}_. : \underline{a}_. | \underline{r}_., \underline{s}_.) = 0$

Zeroing an arrow is the same as deleting it. The 3 rules of Do Calculus can be presented in the same format.

- **Rule 1:** Insertion or deletion of observations

If $(\underline{b}_. \perp \underline{a}_. | \underline{r}_., \underline{s}_.)$ in $D_{\underline{r}_.} G$, then

✓ $P(b.|a., D\underline{r}_. = r., s.) = P(b.|D\underline{r}_. = r., s.) \quad (\text{i.e., } \underline{a}_. = a. \leftrightarrow 1)$



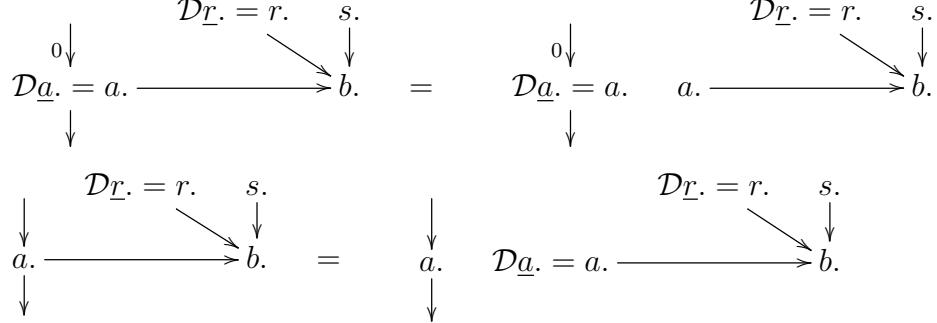
✓ $H(\underline{b}_. : \underline{a}_. | D\underline{r}_., \underline{s}_.) = 0$

³In Statistics, one says a probability distribution $P(x; \theta)$ of x that depends on a parameter θ is **identifiable** if $P(x; \theta_1) = P(x; \theta_2)$ implies $\theta_1 = \theta_2$.

- **Rule 2:** Action or observation exchange

If $(\underline{b} \perp \underline{a} | \underline{r}, \underline{s})$ in $\mathcal{L}_{\underline{a}} \mathcal{D}_{\underline{r}} G$, then

$$\checkmark P(b | \mathcal{D}\underline{a}. = a., \mathcal{D}\underline{r}. = r., s.) = P(b | a., \mathcal{D}\underline{r}. = r., s.) \quad (\text{i.e., } \mathcal{D}\underline{a}. = a. \leftrightarrow a. = a.)$$



In this rule, the node is split into a $\mathcal{D}\underline{a}. = a.$ node and a $a.$ node. The original node keeps the arrows, and the new node is a root node.

$$\checkmark H(\underline{b}. : \mathcal{D}\underline{a}. | \mathcal{D}\underline{r}, \underline{s}) = H(\underline{b}. : \underline{a}. | \mathcal{D}\underline{r}, \underline{s})$$

- **Rule 3:** Insertion and deletion of actions

If $(\underline{b} \perp \underline{a} | \underline{r}, \underline{s})$ in $\mathcal{D}_{\underline{a}-an(\underline{s})} \mathcal{D}_{\underline{r}} G$, then

$$\checkmark P(b | \mathcal{D}\underline{a}. = a., \mathcal{D}\underline{r}. = r., s.) = P(b | \mathcal{D}\underline{r}. = r., s.) \quad (\text{i.e., } \mathcal{D}\underline{a}. = a. \leftrightarrow 1)$$



$$\checkmark H(\underline{b}. : \mathcal{D}\underline{a}. | \mathcal{D}\underline{r}, \underline{s}) = 0$$

See Fig.17.3 for a pictorial representation of these rules.

These rules have been proven to be sufficient for removing all do operators from an expression for which it is possible to do so.

Next we discuss two formulae that can be proven using Do Calculus: the backdoor and the frontdoor adjustment formulae.

The backdoor formula adjusts one multinode and the frontdoor formula adjusts two.

17.2 Parent Adjustment Formula

Suppose that $\underline{x}, \underline{y}, \underline{z}$ are disjoint multinodes and their union equals the totality of all nodes of a bnet. Suppose we have data available that allows us to estimate $P(x., y., z.)$. Hence, all nodes of the bnet are observable. Furthermore, suppose $\underline{z} = pa(\underline{x})$. In other words, we are considering the bnet

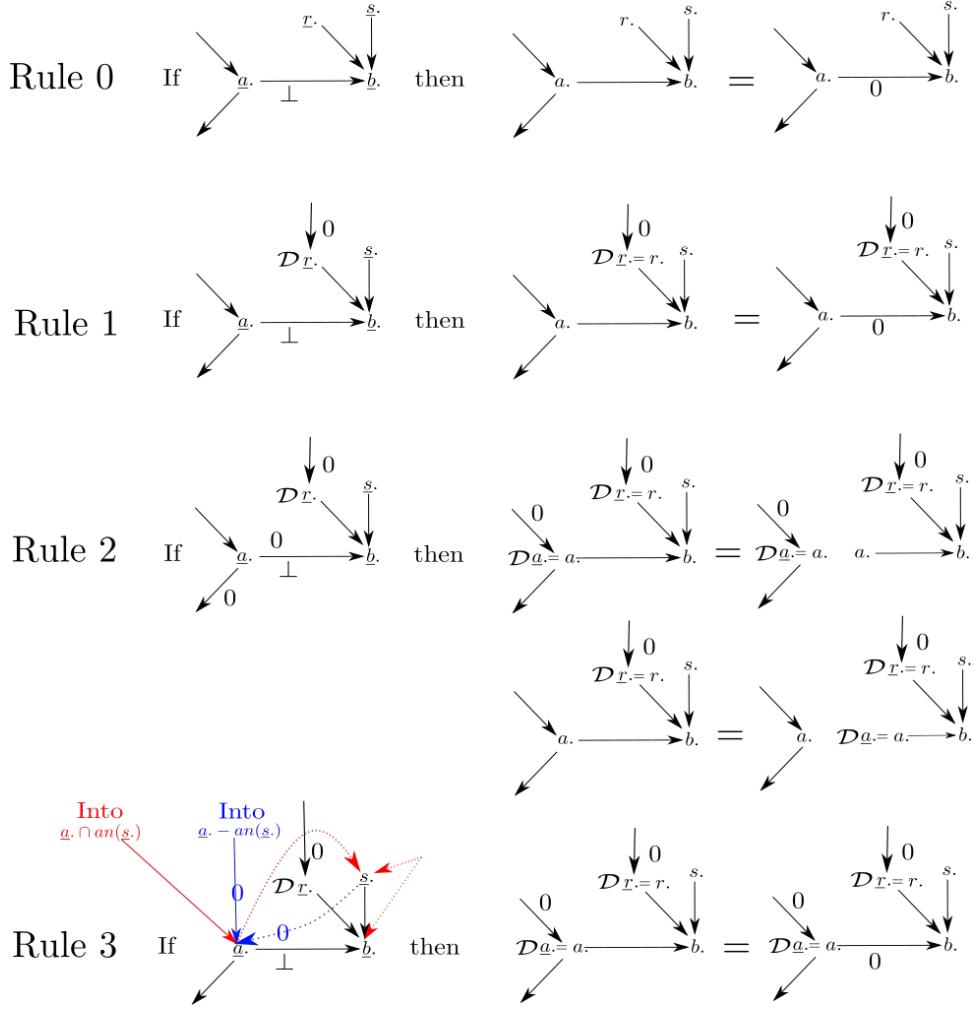


Figure 17.3: Pictorial representation of the rules of Do Calculus. $an(\underline{s.})$ stands for the ancestors of $\underline{s.}$. In Rule 2, we zero the arrows leaving $\underline{a.}$ and ascertain that there is no flow of info between $\underline{a.}$ and $\underline{b.}$ under those circumstances. In Rule 3, as shown by the dotted arrows, conditioning on $\underline{s.}$ is enough to block paths going from $\underline{b.}$ into $\underline{a.} - an(\underline{s.})$, but it is not sufficient to block paths going from $\underline{b.}$ into $\underline{a.} \cap an(\underline{s.})$. In the example with red dotted arrows, conditioning on $\underline{s.}$ opens a path between $\underline{a.}$ and $\underline{b.}$ in which $\underline{s.}$ is a collider. That is the reason for zeroing arrows going into $\underline{a.} \cap an(\underline{s.})$, but not zeroing arrows going into $\underline{a.} \cap an(\underline{s.})$.

$$\begin{array}{c} \underline{z.} \\ \downarrow \\ \underline{x.} \xrightarrow[]{} \underline{y.} \end{array} \quad . \quad (17.19)$$

Then

$$P(y., z. | \mathcal{D}\underline{x}. = x.) = P(y.|x., z.)P(z.) \quad (17.20)$$

so

$$P(y.| \mathcal{D}\underline{x}. = x.) = \sum_{z.} P(y.|x., z.)P(z.) \quad (17.21)$$

This is called **adjusting the parents** of $\underline{x}.$

We say that we are **adjusting or controlling a node \underline{a}** if we condition a probability on \underline{a} and then we average that probability over \underline{a} . More generally, we can adjust a whole multinode \underline{a} . together.

Next, we will introduce a generalization of this parent adjustment formula called the backdoor adjustment formula. In a backdoor adjustment formula, the adjusted multinode is not necessarily the parents of $\underline{x}.$.

17.3 Backdoor Adjustment Formula

See Chapter 3 for examples of the use of the backdoor adjustment formula. In this section, we shall mainly be concerned with proving this theorem using Do Calculus.

For any two disjoint multinodes $\underline{x}.$ and $\underline{y}.$, we define a **backdoor path** from $\underline{x}.$ to $\underline{y}.$ as a path from $\underline{x}.$ and $\underline{y}.$ that starts with an arrow pointing into $\underline{x}.$

Suppose that we have access to data that allows us to estimate a probability distribution $P(x., y., z.).$ Hence, the variables $\underline{x}., \underline{y}., \underline{z}.$ are ALL observed (i.e, not hidden). Then we say that the backdoor $\underline{z}.$ satisfies the **backdoor adjustment criterion** relative to $(\underline{x}., \underline{y}.)$ if

1. All backdoor paths from $\underline{x}.$ to $\underline{y}.$ are blocked by conditioning on $\underline{z}..$
2. $\underline{z}.. \cap de(\underline{x}.) = \emptyset.$

Motivation for BD criterion: Part 1 rules out paths from \underline{x} to \underline{y} containing a fork node (confounder) which, if not blocked by conditioning on $\underline{z}..$, would introduce a non-causal correlation (confounder bias). Part 2 rules out a directed path from \underline{x} to \underline{y} that has a mediator node blocked by conditioning on $\underline{z}..$ or a collider node unblocked by conditioning on $\underline{z}..$

Claim 17 (Backdoor Adjustment Formula)

If $\underline{z}.$ satisfies the backdoor criterion relative to $(\underline{x}., \underline{y}.)$, then

$$P(y.| \mathcal{D}\underline{x}. = x.) = \sum_{z.} P(y.|x., z.)P(z.) \quad (17.22)$$

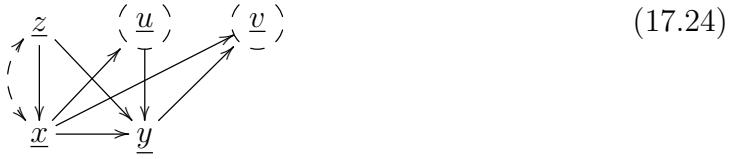
$$= \sum_{z.} \begin{array}{c} \nearrow \\ z. \end{array} \quad (17.23)$$

$x. \longrightarrow y.$

where $\sum z.$ means node $\underline{z}.$ is summed over.

proof:

For simplicity, let us omit the dots from the multinodes. If z satisfies the backdoor criterion relative to $(\underline{x}, \underline{y})$, then $\underline{x}, \underline{y}, \underline{z}$ might have the following structure.



See Claim 21 for a proof of this claim for the special case Eq.(17.24).

QED

Note that the backdoor adjustment formula can be written as

$$P(y. | \mathcal{D}\underline{x}. = x.) = \sum_{z.} P(y.|x., z.)P(z.) \quad (17.25)$$

$$= \sum_{z.} \frac{P(y., x., z.)}{P(x.|z.)} \quad (17.26)$$

This assumes $P(x.|z.) \neq 0$ for all $x., z..$ This assumption is referred to as **positivity**, and is violated if $P(x.|z.) = \delta(x., x.(z.))$. $P(x.|z.)$ is called the **propensity score** of $x.$ given $z..$ This equation does **inverse probability weighting**. One can approximate $P(x.|z.)$ in this equation to get an approximation to $P(y|\mathcal{D}\underline{x} = x)$.

17.4 Frontdoor Adjustment Formula

See Chapter 25 for examples of the use of the frontdoor adjustment formula. In this section, we shall mainly be concerned with proving this theorem using Do Calculus.

Suppose that we have access to data that allows us to estimate a probability distribution $P(x., m., y.).$ Hence, the variables $\underline{x}, \underline{m}, \underline{y}$ are ALL observed (i.e, not hidden). Then we say that the frontdoor $\underline{m}.$ satisfies the **frontdoor adjustment criterion** relative to $(\underline{x}, \underline{y})$ if

1. All directed paths from $\underline{x}.$ to $\underline{y}.$ are intercepted by (i.e., have a node in) $\underline{m}..$
2. All backdoor paths from $\underline{x}.$ to $\underline{m}.$ are blocked.
3. All backdoor paths from on $\underline{m}.$ to $\underline{y}.$ are blocked by conditioning on $\underline{x}..$

Claim 18 (Frontdoor Adjustment Formula)

If $\underline{m}.$ satisfies the frontdoor criterion relative to $(\underline{x}, \underline{y})$, and $P(x., m.) > 0,$ then

$$P(y_{\cdot} | \mathcal{D}\underline{x}_{\cdot} = x_{\cdot}) = \sum_{m_{\cdot}} \underbrace{\left[\sum_{x'_{\cdot}} P(y_{\cdot} | x'_{\cdot}, m_{\cdot}) P(x'_{\cdot}) \right]}_{P(y_{\cdot} | \mathcal{D}\underline{m}_{\cdot} = m_{\cdot})} \underbrace{P(m_{\cdot} | x_{\cdot})}_{P(m_{\cdot} | \mathcal{D}\underline{x}_{\cdot} = x_{\cdot})} \quad (17.27)$$

$$= \sum_{x'_{\cdot}} \sum_{m_{\cdot}} \xrightarrow{x_{\cdot} \longrightarrow \sum m_{\cdot} \longrightarrow y_{\cdot}} \quad (17.28)$$

where $\sum x'_{\cdot}$. and $\sum m_{\cdot}$ means nodes \underline{x}'_{\cdot} . and \underline{m}_{\cdot} are summed over.

proof:

For simplicity, let us omit the dots from the multinodes. If \underline{m} satisfies the frontdoor criterion relative to $(\underline{x}, \underline{y})$, then $\underline{x}, \underline{m}, \underline{y}$ might have the following structure, where node \underline{c} is unobserved.



See Claim 22 for a proof of this claim for the special case Eq.(17.29).

See also Ref.[34] for a proof by Pearl of the Frontdoor Adjustment Formula without using Do Calculus.

QED

17.5 Comparison of Backdoor and Frontdoor adjustment formulae

Define a **direct effect path** for a query $P(y | \mathcal{D}\underline{x} = x, z_{\cdot})$ as a directed path that starts at \underline{x} and ends at \underline{y} . A backdoor path (i.e., one that connects \underline{x} and \underline{y} starting with an arrow pointing into \underline{x}), is not a direct effect path; it's an **indirect effect path**.

Note that in the backdoor AF (adjustment formula), we can find a possibly empty observed multinode \underline{z}_{\cdot} such that if we condition on \underline{z}_{\cdot} , all indirect effect paths are blocked. In the frontdoor AF, we can't find a multinode \underline{z}_{\cdot} that blocks all indirect effect paths. Despite this, in the frontdoor scenario, the do-query is identifiable and an adjustment formula exists. How is that possible? The frontdoor AF uses the backdoor AF once and then it uses the backdoor AF again, a second time, on the result of the first use. The frontdoor AF replaces a sum over an unobserved node by a sum over an observed one.

17.6 Do operator for DEN diagrams

Recall that the structural equations for a linear DEN, as given by Eq.(38.21) of Chapter 38, are:

$$\underline{x} = A\underline{x} + \underline{u}. \quad (17.30)$$

Therefore,

$$\underline{x} = (1 - A)^{-1}\underline{u} \quad (17.31)$$

which can be represented for both linear and non-linear DEN diagrams by:

$$\underline{x}_i = x_i(\underline{u}) \quad (17.32)$$

If now we apply the operator $\mathcal{D}_{\underline{a}=\underline{a}}$ to the diagram described by the structural equations Eqs.17.30, we get the following new structural equations:

$$\underline{x}_i^* = \begin{cases} \sum_{j < i} A_{i|j} \underline{x}_j^* + \underline{u}_i & \text{if } \underline{x}_i \neq \underline{a} \\ \underline{a} & \text{if } \underline{x}_i = \underline{a} \end{cases}, \quad (17.33)$$

where we are calling \underline{x}_i^* the nodes of the DEN diagram post intervention.

Eqs.(17.33) can be expressed in matrix notation as follows. Define $\pi_{\underline{a}}$ to be the $nx \times nx$ matrix with all entries equal to zero except for the (i_0, i_0) entry, which is 1. And define $e_{\underline{a}}$ to be the column vector with all entries zero except for the i_0 'th one, which is 1. Here i_0 is defined so that $\underline{x}_{i_0} = \underline{a}$. In other words, $\pi_{\underline{a}}$ and $e_{\underline{a}}$ are defined by

$$(\pi_{\underline{a}})_{i,j} = \mathbb{1}(i = j, \underline{a} = \underline{x}_i) \quad (17.34)$$

and

$$(e_{\underline{a}})_i = \mathbb{1}(\underline{a} = \underline{x}_i), \quad (17.35)$$

for $i, j \in \{0, 1, \dots, nx - 1\}$. Next define

$$\pi_{!\underline{a}} = 1 - \pi_{\underline{a}}, \quad (17.36)$$

$$A^* = \pi_{!\underline{a}} A, \quad (17.37)$$

and

$$\underline{u}_{!\underline{a}} = \pi_{!\underline{a}} \underline{u}. \quad (17.38)$$

The effect of pre-multiplying the matrix A and the column vector \underline{u} by $\pi_{!\underline{a}}$ is to leave all rows intact except for the i_0 row, which is set to zero. Here i_0 is defined by $\underline{a} = \underline{x}_{i_0}$.

Finally, using all of the variables just defined, we can express the structural equations of the linear DEN diagram, post intervention, as

$$\underline{x}^* = A^* \underline{x}^* + \underline{u}_{!a} + ae_a . \quad (17.39)$$

Thus,

$$\underline{x}^* = (1 - A^*)^{-1}(\underline{u}_{!a} + ae_a) . \quad (17.40)$$

which can be represented for both linear and non-linear DEN diagrams by:

$$\underline{x}_i^* = x_i^*(\underline{u}_{!a}, a) . \quad (17.41)$$

For any bnet,

$$P(\underline{y} = y | \underline{x} = x) = P_G(\underline{y} = y | \underline{x} = x) \quad (17.42)$$

$$P(\underline{y} = y | \mathcal{D}\underline{x} = x) = P_{\mathcal{D}_{\underline{x}=x}G}(\underline{y} = y) \quad (17.43)$$

Claim 19 *For a non-linear DEN diagram,*

$$P(y | \mathcal{D}\underline{x} = x) = E [\delta[y, y(\underline{u}_{!x}, x)]] . \quad (17.44)$$

proof:

$$P(\underline{y} = y | \mathcal{D}\underline{x} = x) = P_{\mathcal{D}_{\underline{x}=x}G}(\underline{y} = y) \quad (17.45)$$

$$= \sum_{u_{!x}} P(u_{!x}) P_{\mathcal{D}_{\underline{x}=x}G}(\underline{y} = y | u_{!x}) \quad (17.46)$$

$$= \sum_{u_{!x}} P(u_{!x}) \delta[y, y(u_{!x}, x)] \quad (17.47)$$

$$= E_{u_{!x}} [\delta[y, y(u_{!x}, x)]] \quad (17.48)$$

$$= E[\delta[y, y(\underline{u}_{!x}, x)]] \quad (17.49)$$

QED

Claim 20 *For a nonlinear DEN diagram,*

$$E[\underline{y} | \mathcal{D}\underline{x} = x] = E[y(\underline{u}_{!x}, x)] . \quad (17.50)$$

proof:

$$E[\underline{y}|\mathcal{D}\underline{x} = x] = \sum_y y P(\underline{y} = y|\mathcal{D}\underline{x} = x) \quad (17.51)$$

$$= \sum_y y E[\delta[y, y(u_{!\underline{x}}, x)]] \quad (17.52)$$

$$= E[y(u_{!\underline{x}}, x)] \quad (17.53)$$

QED

For any bnet

$$P(y|\mathcal{D}\underline{x} = x, z) = \frac{P(y, z|\mathcal{D}\underline{x} = x)}{P(z|\mathcal{D}\underline{x} = x)} = P_{\mathcal{D}_{\underline{x}=x}G}(y|x, z) \quad (17.54)$$

For a nonlinear DEN diagram,

$$P(y, z|\mathcal{D}\underline{x} = x) = \sum_{u_{!\underline{x}}} P(u_{!\underline{x}}) \delta[y, y(u_{!\underline{x}}, x)] \delta[z, z(u_{!\underline{x}}, x)] \quad (17.55)$$

$$P(z|\mathcal{D}\underline{x} = x) = \sum_{u_{!\underline{x}}} P(u_{!\underline{x}}) \delta[z, z(u_{!\underline{x}}, x)]. \quad (17.56)$$

Chapter 18

Do Calculus proofs

In Chapter 17 of Bayesuvius, we explained Do Calculus but referred to this chapter for proofs of claims that use Do Calculus. In this chapter, we've aggregated all proofs, from throughout the book, of claims that use Do Calculus.

Note that even though the 3 rules of Do Calculus are great for proving adjustment formulae for general classes of DAGs, they are sometimes overkill for proving adjustment formulae for a single specific DAG. After all, the 3 rules of Do Calculus are a consequence of the d-separation theorem. Hence, all adjustment formulae should be provable from first principles, assuming only the d-separation theorem and the standard rules of probability theory.

In this chapter, we use the following conventions.

Random variables are underlined and their values are not. For example, $\underline{a} = a$ means the random variable \underline{a} takes the value a . Diagrams with nodes that are underlined represent Bayesian Networks (bnets) and the same diagram with the letters not underlined represents a specific **instantiation** of that bnet. For example $\underline{a} \rightarrow \underline{b}$ represents the bnet with conditional probability distribution $P(b|a)$, whereas $a \rightarrow b$ represents $P(b|a)$ itself.

If \underline{a} is a root node, then $\sum \underline{a}$ signifies a weighted sum $\sum_a P(a)$. For example, $\sum \underline{a} \rightarrow b = \sum_a P(a)P(b|a)$. If \underline{a} is not a root node as in $x \rightarrow \sum \underline{a} \rightarrow y = \sum_a P(y|a)P(a|x)$, then $\sum \underline{a}$ signifies a simple unweighted sum \sum_a .

Unobserved nodes are indicated by enclosing them in a dashed circle. For example, (\underline{u}) .

Selection diagrams with selection nodes are discussed in Chapter 76. In a selection diagram with a selection node $\underline{s} \in \{0, 1\}$, if a node \underline{x} has parents $pa(\underline{x})$ where $\underline{s} \notin pa(\underline{x})$, then the TPM of \underline{x} is $P(x|pa(x))$. If, on the other hand, \underline{x} has parents $pa(\underline{x}) \cup \underline{s}$, then the TPM of \underline{x} is $P(x|pa(x), \underline{s})$, where $P(x|pa(x), \underline{s} = 0) = P(x|pa(x))$ and $P(x|pa(x), \underline{s} = 1) = P^*(x|pa(x))$.

Some identities that are used in this chapter:

1.

$$P(y|x_1, x_2) = \sum_a P(y|a, x_1, x_2)P(a|x_1, x_2). \quad (18.1)$$

$$\begin{array}{ccc}
x_1 & \searrow & x_1 \\
& \nearrow y = & \swarrow \sum a \rightrightarrows y \\
x_2 & & x_2
\end{array} \quad . \quad (18.2)$$

One can describe this identity as “giving y a universal backdoor”, because $\sum a$ is a backdoor (i.e., input) to y , and $\sum a$ is universal in the sense that it is entered by every arrow that enters y except $\sum a$ itself.

2.

$$\sum_a P(a|x_1, x_2) = 1 \quad (18.3)$$

$$\begin{array}{ccc}
x_1 & \searrow & \sum a \xrightarrow{0} \\
& \nearrow x_2 &
\end{array} \quad = 1 \quad (18.4)$$

One can describe this identity as “summing over the values of a collider node which has no emerging arrows”¹. Eq.(18.4) can be understood as an edge case (when $\underline{y} = \emptyset$) of Eq.(18.2).

3.

$$\sum_a P(x_2|a)P(a|x_1) = P(x_2|x_1) \quad (18.5)$$

$$x_1 \longrightarrow \sum a \longrightarrow x_2 = x_1 \longrightarrow x_2 \quad (18.6)$$

One can describe this identity as “summing over the values of a mediator node”.

4.

$$P(x) = \sum_a P(x|a)P(a) = \sum_b P(x|b)P(b) \quad (18.7)$$

$$P(x) = \xrightarrow{0} \sum a \longrightarrow x = \xrightarrow{0} \sum b \longrightarrow x \quad (18.8)$$

One can describe this identity as “averaging over different priors”. Eq.(18.8) can be understood as an edge case of Eq.(18.6).

¹A zeroed arrow means the same as no arrow.

A **do-adjustment formula** expresses a **do-query** (i.e., a conditional probability with do operators in its condition) by an equivalent expression without do operators. If a do-adjustment formula exists for a particular do-query, then we say the do-query is **do-identifiable**.² See Fig.18.1 for some simple examples of identifiable and non-identifiable do-queries.

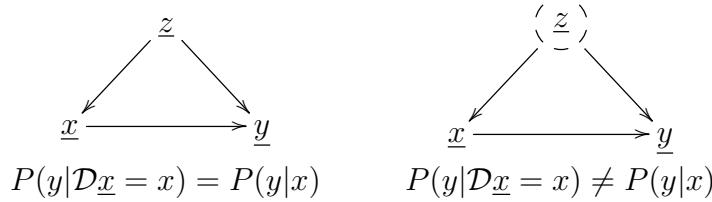


Figure 18.1: Examples of identifiable and non-identifiable do-queries. $\mathcal{D}\underline{x} = x$ in a do-query means we erase the arrow $\underline{z} \rightarrow \underline{x}$ and replace node \underline{x} by a delta function $\delta(x, x')$. Let $Q(y|x) = \sum_z P(y|x, z)P(z)$. In both cases, $P(y|\mathcal{D}\underline{x} = x) = Q(y|x)$, but in the non-identifiable case, $Q(y|x)$ is summing over the unobserved node \underline{z} , so it is using data that is not available. To distinguish between these two possibilities for $Q(y|x)$, we set $P(y|x) = Q(y|x)$ in the identifiable case, and $P(y|\mathcal{D}\underline{x} = x) = Q(y|\mathcal{D}\underline{x} = x)$ in the non-identifiable case.

The following is the simplest of do-adjustment formulae:

$$P(y|\mathcal{D}\underline{x} = x) = P(y|x) \quad (18.9)$$

By definition, since identifiability gives permission us to remove \mathcal{D} operators, all do-identifiable problems satisfy this simplest of adjustment formulae. However, we aim to find an adjustment formula that utilizes the full distribution of the *observed* nodes. Adjustment formulae that don't utilize the full distribution of the observed nodes are throwing away useful info from the dataset, and are less sensitive to deviations from the DAG model being hypothesized.

Given a bnet G and an adjustment formula AF for the query $P(y|\mathcal{D}\underline{x} = x, z)$, a simple identification method (SIM) that we use in this chapter to prove that AF applies to G , is the following:

1. From the bnet G , write an instantiation of G in which the arrows entering node x are amputated, node x is replaced by $\mathcal{D}\underline{x} = x$, and all observed and unobserved nodes, except x, y, z , are summed over.
2. Replace all sums over hidden nodes by sums over observed nodes.

²To prove that a do-query $P(y|\mathcal{D}\underline{x} = x, z)$ is do-identifiable for a graph G , just prove that $y \perp \underline{x}|z$ in $\mathcal{L}_{\underline{x}}G$. This is called Rule 2 of Do Calculus, but it is easy to understand just from the d-separation theorem. Info can be transmitted between y and \underline{x} by either (1) paths in \mathcal{D}_xG or (2) paths in \mathcal{L}_xG . $P(y|\mathcal{D}\underline{x} = x, z) = P(y|x, z)$ means the info is being transmitted only by (1). So the Rule 2 premise is checking that no info is being transmitted by (2).

3. Once all nodes in the diagram represent observed nodes only, we can replace the node $\mathcal{D}\underline{x} = x$ by x . This is justified because in the latest diagram (the one with no unobserved nodes), unavailable info that belongs to unobserved nodes is not being used. After all, that is the meaning of identifiability: that we can express a do-query using only the info that is available.

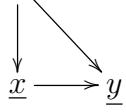
You can check that using SIM, it is not possible to find an adjustment formula for the example of a non-identifiable query in Fig.18.1. SIM does yield the backdoor adjustment formula for the identifiable query in Fig.18.1.

A **do-transport formula** expresses a do-query in terms of an equivalent do-query.

This chapter deals with both do-adjustment and do-transport formulae.

Claim 21 (*Backdoor Adjustment Formula*)

If $\begin{array}{c} z \\ \downarrow \\ \underline{x} \xrightarrow{} y \end{array}$ then



$$P(y|\mathcal{D}\underline{x} = x) = \sum_z P(y|x, z)P(z) \quad (18.10)$$

$$\begin{array}{ccc} \mathcal{D}\underline{x} = x & \longrightarrow & \sum_z \\ & & \searrow \\ & = & x \longrightarrow y \end{array} \quad (18.11)$$

proof:

* **proof 1:**

$$P(y|\mathcal{D}\underline{x} = x) = \sum_z P(y|x, z)P(z)$$

We can replace
 $\mathcal{D}\underline{x} = x$ by x
once all nodes
in bnet are
observed nodes.

$$\begin{array}{ccc} \sum_z & & \sum_z \\ \searrow & = & \searrow \\ \mathcal{D}\underline{x} = x & \longrightarrow & x \longrightarrow y \end{array}$$

* **proof 2:**

$$\begin{aligned} P(y|\mathcal{D}\underline{x} = x) &= \sum_z P(y|\mathcal{D}\underline{x} = x, z)P(z|\mathcal{D}\underline{x} = x) \\ &\text{by Probability Axioms} \\ &= \sum_z P(y|x, z)P(z|\mathcal{D}\underline{x} = x) \\ &P(y|\mathcal{D}\underline{x} = x, z) \rightarrow P(y|x, z) \end{aligned}$$

by Rule 2: If $(\underline{b} \perp \underline{a} | \underline{r}, \underline{s})$ in $\mathcal{L}_{\underline{a}} \mathcal{D}_{\underline{r}} G$, then
 $\mathcal{D}_{\underline{a}} = a \leftrightarrow \underline{a} = a$.

$$\underline{y} \perp \underline{x} | \underline{z} \text{ in } \mathcal{L}_{\underline{x}} \mathcal{D}_{\emptyset} G : \begin{array}{c} \underline{z} \\ \downarrow \\ \underline{x} \end{array} \quad \begin{array}{c} \searrow \\ \underline{y} \end{array}$$

$$= \sum_{\underline{z}} P(y|\underline{x}, \underline{z}) P(\underline{z}) \\ P(z|\mathcal{D}_{\underline{x}} = x) \rightarrow P(z)$$

by Rule 3: If $(\underline{b} \perp \underline{a} | \underline{r}, \underline{s})$ in $\mathcal{D}_{\underline{a} - an(\underline{s})} \mathcal{D}_{\underline{r}} G$, then
 $\mathcal{D}_{\underline{a}} = a \leftrightarrow 1$

$$\underline{z} \perp \underline{x} \text{ in } \mathcal{D}_{\underline{x}} \mathcal{D}_{\emptyset} G : \begin{array}{c} \underline{z} \\ \searrow \\ \underline{x} \end{array} \longrightarrow \underline{y}$$

QED

Claim 22 (*Frontdoor Adjustment Formula*)

$$\text{If } \begin{array}{c} (\bar{c}) \\ \diagup \quad \diagdown \\ \underline{x} \quad \underline{m} \end{array} \text{ then } \begin{array}{c} \underline{m} \longrightarrow \underline{y} \end{array}$$

$$P(y|\mathcal{D}_{\underline{x}} = x) = \sum_m \left[\sum_{x'} P(y|x', m) P(x') \right] P(m|x) \quad (18.12)$$

$$\mathcal{D}_{\underline{x}} = x \longrightarrow y \quad \begin{array}{c} \sum x' \\ \diagup \quad \diagdown \\ x \longrightarrow \sum m \longrightarrow y \end{array} \quad (18.13)$$

proof:

* **proof 1:**

$$P(y|\mathcal{D}_{\underline{x}} = x) = \sum_{m,c,x'} P(y|m, c) P(c|x') P(x') P(m|\mathcal{D}_{\underline{x}} = x)$$

$$\begin{array}{ccc} \begin{array}{c} (\sum c) \\ \diagup \quad \diagdown \\ \mathcal{D}_{\underline{x}} = x \longrightarrow \sum m \longrightarrow y \end{array} & = & \begin{array}{c} \sum x' \longrightarrow (\sum c) \\ \diagup \quad \diagdown \\ \mathcal{D}_{\underline{x}} = x \longrightarrow \sum m \longrightarrow y \end{array} \end{array}$$

$$= \sum_{m,x'} P(y|m, x') P(x') P(m|x)$$

We can replace
 $\mathcal{D}\underline{x} = x$ by x
once all nodes
in bnet are
observed nodes.

$$\begin{array}{c} \sum x' \\ \searrow \\ = \quad x \longrightarrow \sum m \longrightarrow y \end{array}$$

* proof 2:

$$P(y|\mathcal{D}\underline{x} = x) = \sum_m P(y|\mathcal{D}\underline{x} = x, m) P(m|\mathcal{D}\underline{x} = x)$$

by Probability Axioms

$$= \sum_m P(y|\mathcal{D}\underline{x} = x, \mathcal{D}\underline{m} = m) P(m|\mathcal{D}\underline{x} = x)$$

$$P(y|\mathcal{D}\underline{x} = x, m) \rightarrow P(y|\mathcal{D}\underline{x} = x, \mathcal{D}\underline{m} = m)$$

by Rule 2: If $(\underline{b} \perp \underline{a} | \underline{r}, \underline{s})$ in $\mathcal{L}_{\underline{a}} \mathcal{D}_{\underline{r}} G$, then
 $\mathcal{D}\underline{a} = a \leftrightarrow \underline{a} = a$.

$$\underline{y} \perp \underline{m} | \underline{x} \text{ in } \mathcal{L}_{\underline{m}} \mathcal{D}_{\underline{x}} G : \quad \begin{pmatrix} \overset{\text{c}}{\diagup} \\ \diagdown \end{pmatrix} \quad \begin{matrix} \underline{x} \longrightarrow \underline{m} \\ \longrightarrow \underline{y} \end{matrix}$$

$$= \sum_m P(y|\mathcal{D}\underline{x} = x, \mathcal{D}\underline{m} = m) P(m|x)$$

$$P(m|\mathcal{D}\underline{x} = x) \rightarrow P(m|x)$$

by Rule 2: If $(\underline{b} \perp \underline{a} | \underline{r}, \underline{s})$ in $\mathcal{L}_{\underline{a}} \mathcal{D}_{\underline{r}} G$, then
 $\mathcal{D}\underline{a} = a \leftrightarrow \underline{a} = a$.

$$\underline{m} \perp \underline{x} \text{ in } \mathcal{L}_{\underline{x}} \mathcal{D}_{\emptyset} G : \quad \begin{pmatrix} \overset{\text{c}}{\diagup} \\ \diagdown \end{pmatrix} \quad \begin{matrix} \underline{x} \\ \diagdown \\ \underline{m} \longrightarrow \underline{y} \end{matrix}$$

$$= \sum_m P(y|\mathcal{D}\underline{m} = m) P(m|x)$$

$$P(y|\mathcal{D}\underline{x} = x, \mathcal{D}\underline{m} = m) \rightarrow P(y|\mathcal{D}\underline{m} = m)$$

by Rule 3: If $(\underline{b} \perp \underline{a} | \underline{r}, \underline{s})$ in $\mathcal{D}_{\underline{a}-an(\underline{s})} \mathcal{D}_{\underline{r}} G$, then
 $\mathcal{D}\underline{a} = a \leftrightarrow 1$

$$\underline{y} \perp \underline{x} | \underline{m} \text{ in } \mathcal{D}_{\underline{x}} \mathcal{D}_{\underline{m}} G : \quad \begin{pmatrix} \overset{\text{c}}{\diagup} \\ \diagdown \end{pmatrix} \quad \begin{matrix} \underline{x} \\ \diagdown \\ \underline{m} \longrightarrow \underline{y} \end{matrix}$$

$$= \sum_{x'} \sum_m P(y|\mathcal{D}\underline{m} = m, x') P(x'|\mathcal{D}\underline{m} = m) P(m|x)$$

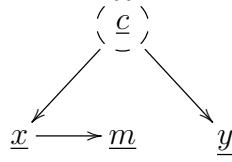
by Probability Axioms

$$= \sum_{x'} \sum_m P(y|m, x') P(x'|\mathcal{D}\underline{m} = m) P(m|x)$$

$$P(y|\mathcal{D}\underline{m} = m, x') \rightarrow P(y|m, x')$$

by Rule 2: If $(\underline{b} \perp \underline{a} | \underline{r}, \underline{s})$ in $\mathcal{L}_{\underline{a}} \mathcal{D}_{\underline{r}} G$, then
 $\mathcal{D}_{\underline{a}} = a \leftrightarrow \underline{a} = a$.

$\underline{y} \perp \underline{m} | \underline{x}$ in $\mathcal{L}_{\underline{m}} \mathcal{D}_{\emptyset} G$:

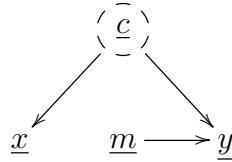


$$= \sum_{x'} \sum_m P(y|m, x') P(x') P(m|x)$$

$$P(x'|\mathcal{D}\underline{m} = m) \rightarrow P(x')$$

by Rule 3: If $(\underline{b} \perp \underline{a} | \underline{r}, \underline{s})$ in $\mathcal{D}_{\underline{a}-an(\underline{s})} \mathcal{D}_{\underline{r}} G$, then
 $\mathcal{D}_{\underline{a}} = a \leftrightarrow 1$

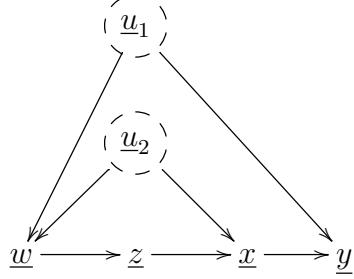
$\underline{x} \perp \underline{m}$ in $\mathcal{D}_{\underline{m}} \mathcal{D}_{\emptyset} G$:



QED

Claim 23 (*Napkin problem from Ref.[43]*)

If \underline{u}_1 then



$$P(y|\mathcal{D}\underline{x} = x) = \sum_{w,z} P(y|x, w, z) P(w, z) \quad (18.14)$$

$$\mathcal{D}\underline{x} = x \longrightarrow y = \sum_{w,z} \begin{matrix} & w, z \\ & \diagdown \\ x & \longrightarrow & y \end{matrix} \quad (18.15)$$

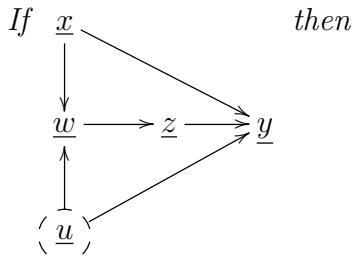
proof:

$$P(y|\mathcal{D}\underline{x} = x) = \sum_{u_1} P(y|\mathcal{D}\underline{x} = x, u_1) P(u_1)$$

$$\begin{aligned}
& \text{Diagram showing a bnet structure: } \\
& \quad \text{Top node: } (\sum u_1) \text{ (solid), } (\sum u_2) \text{ (dashed)} \\
& \quad \text{Bottom nodes: } \sum w \rightarrow \sum z \text{ (solid), } \mathcal{D}\underline{x} = x \rightarrow \underline{y} \text{ (dashed)} \\
& \quad \text{Equation: } = \sum_{w,z} \sum_{u_1} P(y|\mathcal{D}\underline{x} = x, u_1) P(u_1|w, z) P(w, z) \\
& \quad \text{Diagram: } \sum w, z \rightarrow (\sum u_1) \text{ (solid), } \mathcal{D}\underline{x} = x \rightarrow \underline{y} \text{ (dashed)} \\
& \quad \text{Equation: } = \sum_{w,z} P(y|x, w, z) P(w, z) \\
& \quad \text{Text: We can replace } \mathcal{D}\underline{x} = x \text{ by } x \text{ once all nodes in bnet are observed nodes.} \\
& \quad \text{Bottom diagram: } \sum w, z \rightarrow x \rightarrow y \text{ (solid)}
\end{aligned}$$

QED

Claim 24 (from Ref.[43])

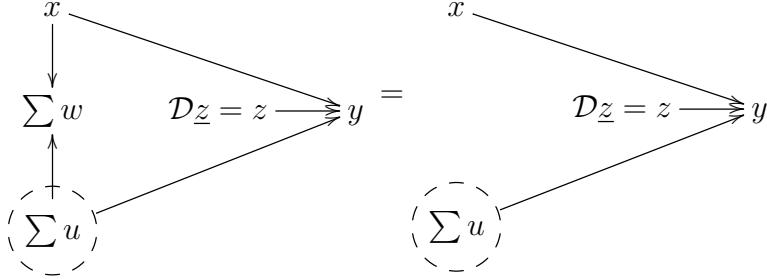


$$P(y|\mathcal{D}\underline{z} = z, x) = \sum_w P(y|z, x, w) P(w) \quad (18.16)$$

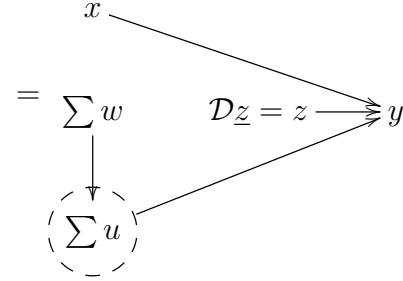
$$\begin{array}{ccc}
x & & x \\
& \searrow & = & \searrow \\
& \mathcal{D}\underline{z} = z \rightarrow y & & \sum w \curvearrowright z \rightarrow y
\end{array} \quad (18.17)$$

proof:

$$P(y|\mathcal{D}\underline{z} = z, x) = \sum_u P(y|\mathcal{D}\underline{z} = z, x, u)P(u)$$

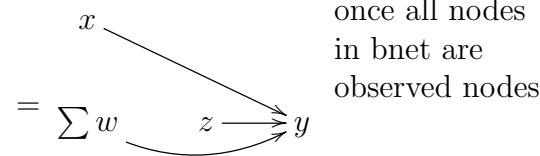


$$= \sum_w \sum_u P(y|\mathcal{D}\underline{z} = z, x, u)P(u|w)P(w)$$



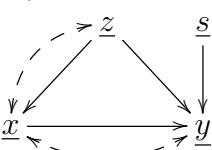
$$= \sum_w P(y|z, x, w)P(w)$$

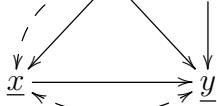
We can replace
 $\mathcal{D}\underline{z} = z$ by z
once all nodes
in bnet are
observed nodes.



QED

Claim 25 (*Trivial Memoryless Transportability, from Ref.[41]*)

If  where $\underline{s} \in \{0, 1\}$ is a selection node, then



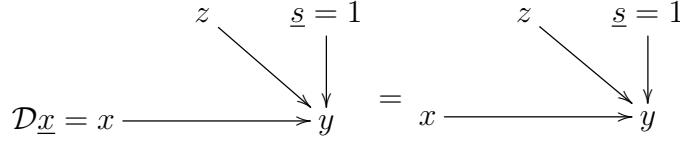
$$P^*(y|\mathcal{D}\underline{x} = x, z) = P^*(y|x, z) \quad (\text{replace } \mathcal{D} \text{ by } 1, \text{ keep } P^*) \quad (18.18)$$

The diagram illustrates the transportability claim for a selection node. It shows two equivalent directed acyclic graphs separated by an equals sign. On the left, there is a node z at the top with an outgoing edge to a node $\underline{s} = 1$ below it. The node $\underline{s} = 1$ has an outgoing edge to a node y below it. There is also a node $\mathcal{D}\underline{x} = x$ at the bottom left with an outgoing edge to y . On the right, there is a node z at the top with an outgoing edge to a node y below it. There is also a node x at the bottom left with an outgoing edge to y .

$$\mathcal{D}\underline{x} = x \longrightarrow y = x \longrightarrow y \quad (18.19)$$

proof:

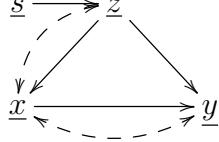
$$P(y|\mathcal{D}\underline{x} = x, z, \underline{s} = 1) = P(y|x, z, \underline{s} = 1)$$



QED

Claim 26 (*Direct Transportability, a.k.a. External Validity, from Ref.[41]*)

If $\underline{s} \xrightarrow{\quad} \underline{z}$ where $\underline{s} \in \{0, 1\}$ is a selection node, then



$$P^*(y|\mathcal{D}\underline{x} = x, z) = P(y|\mathcal{D}\underline{x} = x, z) \quad (\text{replace } P^* \text{ by } P, \text{ keep } \mathcal{D}) \quad (18.20)$$

$$\begin{array}{ccc} \underline{s} = 1 \longrightarrow z & & z \\ \swarrow & \searrow & \downarrow \\ \mathcal{D}\underline{x} = x \longrightarrow y & = & \mathcal{D}\underline{x} = x \longrightarrow y \end{array} \quad (18.21)$$

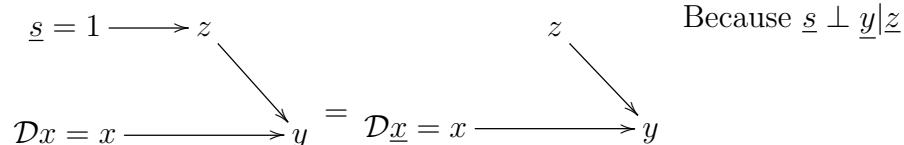
Furthermore,

$$P^*(y|\mathcal{D}\underline{x} = x) = \sum_z P(y|\mathcal{D}\underline{x} = x, z)P^*(z) \quad (18.22)$$

$$\begin{array}{ccc} \underline{s} = 1 & & \underline{s} = 1 \longrightarrow \sum z \\ \searrow & & \swarrow \\ \mathcal{D}\underline{x} = x \longrightarrow y & = & \mathcal{D}\underline{x} = x \longrightarrow y \end{array} \quad (18.23)$$

proof:

$$P(y|\mathcal{D}\underline{x} = x, z, \underline{s} = 1) = P(y|\mathcal{D}\underline{x} = x, z)$$



Furthermore,

$$P(y|\mathcal{D}\underline{x} = x, \underline{s} = 1) = \sum_z P(y|\mathcal{D}\underline{x} = x, z)P(z|\underline{s} = 1)$$

$\underline{s} = 1 \longrightarrow \sum z$

 $\mathcal{D}\underline{x} = x \longrightarrow y$

QED

Claim 27 (*S-Admissible Transportability, from Ref.[41]*)

If $\underline{s} \xrightarrow{\text{---}} (\underline{z}) \xrightarrow{\text{---}} \underline{a}$ where $\underline{s} \in \{0, 1\}$ is a selection node, then

$\underline{x} \xrightarrow{\text{---}} \underline{y}$

$$P^*(y|\mathcal{D}\underline{x} = x) = \sum_a P(y|\mathcal{D}\underline{x} = x, a)P^*(a) \quad (18.24)$$

$$\begin{array}{ccc} \underline{s} = 1 & & \underline{s} = 1 \longrightarrow \sum a \\ \searrow & & \downarrow \\ \mathcal{D}\underline{x} = x \longrightarrow y & = & \mathcal{D}\underline{x} = x \longrightarrow y \end{array} \quad (18.25)$$

proof:

$$P(y|\mathcal{D}\underline{x} = x, \underline{s} = 1) = \sum_a P(y|\mathcal{D}\underline{x} = x, a)P(a|\underline{s} = 1)$$

$\underline{s} = 1 \longrightarrow (\sum z) \longrightarrow \sum a$

 $\mathcal{D}\underline{x} = x \longrightarrow y$

QED

Claim 28 (*Non-transportability, from Ref.[41]*)

If $\underline{s} \xrightarrow{\text{---}} (\underline{h}) \xrightarrow{\text{---}} \underline{s}$ where $\underline{s} \in \{0, 1\}$ is a selection node, then

$\underline{x} \longrightarrow \underline{y}$

$$P^*(y|\mathcal{D}\underline{x} = x) = P^*(y|\mathcal{D}\underline{x} = x) \quad (18.26)$$

$$\begin{array}{ccc}
 & \underline{s} = 1 & \\
 & \downarrow & \\
 \mathcal{D}\underline{x} = x & \xrightarrow{\quad\quad\quad} & y = same
 \end{array} \tag{18.27}$$

proof:

$$P^*(y|\mathcal{D}\underline{x} = x) = P^*(y|\mathcal{D}\underline{x} = x)$$

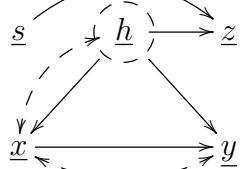
$$\begin{array}{ccc}
 & \left(\sum h \right) & \underline{s} = 1 \\
 & \swarrow \searrow & \downarrow \\
 \mathcal{D}\underline{x} = x & \xrightarrow{\quad\quad\quad} & y = \mathcal{D}\underline{x} = x \xrightarrow{\quad\quad\quad} y \quad \underline{s} = 1
 \end{array}$$

Can't replace $\mathcal{D}\underline{x} = x$ by x because $y \not\propto x$ in $\mathcal{L}_x G$. Hence, Rule 2 not satisfied.

QED

Claim 29 (from Ref.[41])

If \underline{s}  $\xrightarrow{\sum h} z$ where $\underline{s} \in \{0, 1\}$ is a selection node, then



$$P^*(y|\mathcal{D}\underline{x} = x) = P(y|\mathcal{D}\underline{x} = x) \tag{18.28}$$

$$\underline{s} = 1 \quad \mathcal{D}\underline{x} = x \xrightarrow{\quad\quad\quad} y = \mathcal{D}\underline{x} = x \xrightarrow{\quad\quad\quad} y \tag{18.29}$$

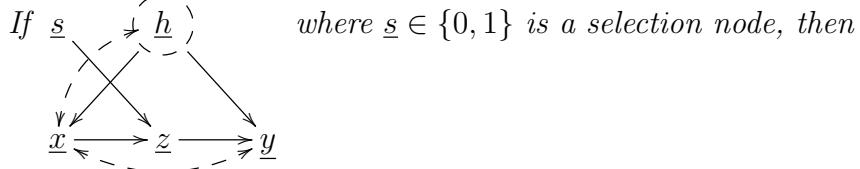
proof:

$$P(y|\mathcal{D}\underline{x} = x, \underline{s} = 1) = \sum_h P(y|\mathcal{D}\underline{x} = x, h)P(h)$$

$$\begin{array}{ccc}
 & \left(\sum h \right) \xrightarrow{\quad\quad\quad} \sum z & \\
 & \swarrow \searrow & \\
 \mathcal{D}\underline{x} = x & \xrightarrow{\quad\quad\quad} & y = \mathcal{D}\underline{x} = x \xrightarrow{\quad\quad\quad} y \\
 = P(y|\mathcal{D}\underline{x} = x) & & \\
 = \mathcal{D}\underline{x} = x \xrightarrow{\quad\quad\quad} y & &
 \end{array}$$

QED

Claim 30 (from Ref.[41])

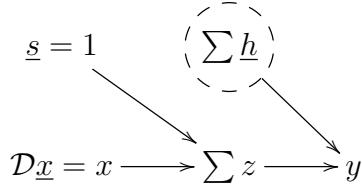


$$P^*(y|\mathcal{D}\underline{x} = x) = \sum_z P(y|\mathcal{D}\underline{x} = x, z)P^*(z|x) \quad (18.30)$$

$$\begin{array}{ccc} \underline{s} = 1 & \mathcal{D}\underline{x} = x & \underline{s} = 1 \\ \searrow & \downarrow & \searrow \\ y & = & x \longrightarrow \sum z \longrightarrow y \end{array} \quad (18.31)$$

proof:

$$P(y|\mathcal{D}\underline{x} = x, \underline{s} = 1) = \sum_h \sum_z P(y|h, z)P(h)P(z|\mathcal{D}\underline{x} = x, \underline{s} = 1)$$

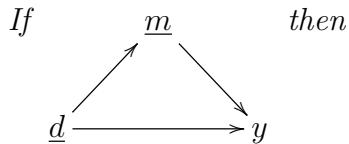


$$= \sum_h \sum_z P(y|h, z)P(h|\mathcal{D}\underline{x} = x)P(z|x, \underline{s} = 1)$$

$$\begin{aligned} & \underline{s} = 1 \quad (\sum h) \quad \mathcal{D}\underline{x} = x \\ & \searrow \qquad \qquad \swarrow \qquad \qquad \downarrow \\ & \mathcal{D}\underline{x} = x \longrightarrow \sum z \longrightarrow y \\ & = \sum_z P(y|\mathcal{D}\underline{x} = x, z)P(z|x, \underline{s} = 1) \\ & = \underline{s} = 1 \quad \mathcal{D}\underline{x} = x \\ & \searrow \qquad \qquad \downarrow \\ & x \longrightarrow \sum z \longrightarrow y \end{aligned}$$

QED

Claim 31 (Unconfounded Mediation, from Ref.[40])



$$P(y|\mathcal{D}\underline{d} = d, \mathcal{I}\underline{m}\underline{d} = d') = \sum_m P(y|d, m)P(m|d') \quad (18.32)$$

$$\begin{array}{ccc} \mathcal{I}\underline{d} = d' & & \mathcal{I}\underline{d} = d' \longrightarrow \sum m \\ \searrow & & \swarrow \\ \mathcal{D}\underline{d} = d \longrightarrow y & = & \mathcal{D}\underline{d} = d \longrightarrow y \end{array} \quad (18.33)$$

proof:

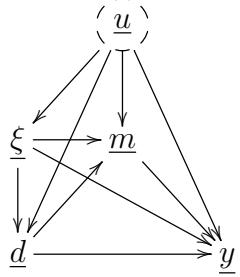
$$P(y|\mathcal{D}\underline{d} = d, \mathcal{I}\underline{d} = d') = \sum_m P(y|d, m)P(m|d')$$

$$\begin{array}{ccc} \mathcal{I}\underline{d} = d' \longrightarrow \sum m & & \\ \searrow & & \\ \mathcal{D}\underline{d} = d \longrightarrow y & & \end{array}$$

QED

Claim 32 (*Mediation with universal prior ξ and universal confounder \underline{u} , from Ref.[40]*)

If ξ then



$$P(y|\mathcal{D}\underline{d} = d, \mathcal{I}\underline{m}\underline{d} = d') = \sum_{\xi} \sum_m P(y|d, m, \xi)P(m|d', \xi)P(\xi) \quad (18.34)$$

$$\begin{array}{ccc} \mathcal{I}\underline{d} = d' & & \mathcal{I}\underline{d} = d' \xrightarrow{\sum \xi} \sum m \\ \searrow & & \swarrow \\ \mathcal{D}\underline{d} = d \longrightarrow y & = & \mathcal{D}\underline{d} = d \longrightarrow y \end{array} \quad (18.35)$$

proof:

$$P(y|\mathcal{D}\underline{d} = d, \mathcal{I}\underline{d} = d') = \sum_{\xi,u} \sum_m P(y|d, m, \xi, u)P(m|d', \xi, u) \underbrace{P(\xi|u)P(u)}_{P(\xi,u)}$$

$$\begin{array}{c}
(\sum u) \\
+ \\
\downarrow \\
\sum \xi \longrightarrow \sum m \\
\downarrow \\
\mathcal{I}\underline{d} = d' \quad \text{---} \quad \sum \xi \longrightarrow \sum m \\
\downarrow \\
\mathcal{D}\underline{d} = d - \sum_{\xi} \sum_m P(y|d, m, \xi)P(m|d', \xi)P(\xi) \\
\longrightarrow y
\end{array}$$

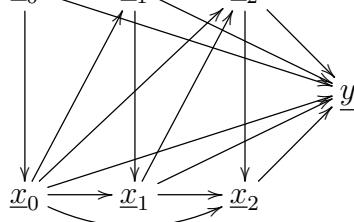
$$\begin{array}{c}
\mathcal{I}\underline{d} = d' \quad \sum \xi \longrightarrow \sum m \\
\downarrow \\
= \mathcal{D}\underline{d} = d \longrightarrow y
\end{array}$$

We switch from averaging over the prior of ξ, u to averaging over the prior of ξ .

QED

Claim 33 (*Sequential backdoor adjustment formula, from Ref.[44]*)

If $\underline{z}_0 \rightarrow \underline{z}_1 \rightarrow \underline{z}_2$ then



$$P(y|\mathcal{D}\underline{x}^3 = x^3) = Q(y|x^3) \quad (18.36)$$

$$\sum z_0 \rightarrow \sum z_1 \rightarrow \sum z_2 \quad (18.37)$$

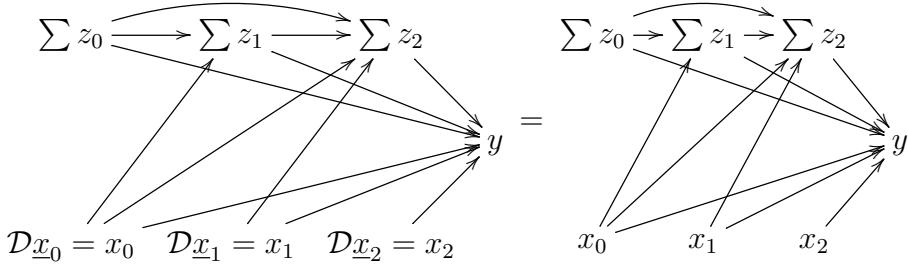
$$\mathcal{D}\underline{x}^3 = x^3 \longrightarrow y =$$



The result shown here for $n = 3$ is true for any integer $n \geq 1$.

proof:

$$P(y|\mathcal{D}\underline{x}^3 = x^3) = Q(y|x^3)$$



We can replace $\mathcal{D}\underline{x}_i = x_i$ by x_i once all nodes in bnet are observed nodes.

QED

Claim 34 (*Selection Bias (SB) Backdoor Adjustment Formula, from Ref.[1]*)

If $\underline{s} \xrightarrow{\quad} \underline{z}^{<\underline{x}} \xrightarrow{\quad} \underline{z}^{>\underline{x}}$ where $\underline{s} \in \{0, 1\}$ is a selection node and $\underline{z} = (\underline{z}^{<\underline{x}}, \underline{z}^{>\underline{x}})$,

then

$$P(y|\mathcal{D}\underline{x} = x, \underline{s} = 1) = \sum_z P(y|x, z)P(z) = P(y|x) \quad (18.38)$$

$$\begin{array}{ccc} \underline{s} = 1 & & \sum z \\ \searrow & & \downarrow \\ \mathcal{D}\underline{x} = x \longrightarrow y & = & x \longrightarrow y \end{array} = x \longrightarrow y \quad (18.39)$$

proof:

$$P(y|\mathcal{D}\underline{x} = x, \underline{s} = 1) = \sum_z P(y|\mathcal{D}\underline{x} = x, z)P(z^{<\underline{x}}|\underline{s} = 1)P(z^{>\underline{x}}|x, z^{<\underline{x}}, \underline{s} = 1)$$

$$\begin{aligned} & \underline{s} = 1 \xrightarrow{\quad} \sum z^{<\underline{x}} \xrightarrow{\quad} \sum z^{>\underline{x}} \\ & \mathcal{D}\underline{x} = x \longrightarrow y \\ &= \sum_{z^{<\underline{x}}} P(y|\mathcal{D}\underline{x} = x, z^{<\underline{x}})P(z^{<\underline{x}}|\underline{s} = 1) \\ & \underline{s} = 1 \longrightarrow \sum z^{<\underline{x}} \\ &= \mathcal{D}\underline{x} = x \longrightarrow y \\ &= \sum_z P(y|x, z)P(z|\underline{s} = 1) \end{aligned}$$

$$\begin{aligned}
 & \underline{s} = 1 \longrightarrow \sum z & \mathcal{D} \text{ can be removed because there are} \\
 & = x \longrightarrow y & \text{no sums over unobserved nodes.} \\
 & = \sum_z P(y|x, z)P(z) & \underline{s} = 1 \text{ node can be removed} \\
 & & \text{because this expression must} \\
 & \sum z & \text{equal } P(y|x, \underline{s} = 1). \text{ Furthermore,} \\
 & & \underline{y} \perp \underline{s} | (\underline{x}, \underline{z}) \text{ in the hypothesis bnet.} \\
 & = x \longrightarrow y & \text{Hence, this expression must also} \\
 & & \text{equal } P(y|x).
 \end{aligned}$$

QED

Chapter 19

D-Separation

Before reading this chapter, I recommend that you read Chapter Definition of a Bayesian Network.

A path γ that isn't a loop can have 3 types of intermediate nodes \underline{x} (an intermediate node of γ is a node in γ that isn't one of the two end nodes). Suppose $\underline{a}, \underline{b} \in \gamma$ are the two neighbors of \underline{x} . Then the 3 possible cases are:

1. **\underline{x} is a mediator node:** $(\underline{a} \leftarrow \underline{x} \leftarrow \underline{b})$ or $(\underline{a} \rightarrow \underline{x} \rightarrow \underline{b})$
2. **\underline{x} is a fork node:** $(\underline{a} \leftarrow \underline{x} \rightarrow \underline{b})$
3. **\underline{x} is a collider node:** $(\underline{a} \rightarrow \underline{x} \leftarrow \underline{b})$

We say that a non-loop path γ from \underline{a} to \underline{b} (i.e., with end nodes $\underline{a}, \underline{b}$) is **blocked** by conditioning on a multinode \underline{Z} . if one or more of the following statements is true:

1. There is a node $\underline{x} \in \underline{Z}$. which is a mediator or a fork of γ .
2. γ contains a collider node \underline{c} and $(\underline{c} \cup de(\underline{c})) \cap \underline{Z} = \emptyset$ (i.e., neither \underline{c} nor any of the descendants of \underline{c} is contained in \underline{Z} .)

This definition of a blocked path¹ is easy to remember if one thinks of the following analogy with pipes carrying a fluid. Think of path γ as if it were a pipe carrying a fluid. Think of the nodes of γ as junctions in the pipe. If \underline{Z} intersects γ at either a mediator or a fork junction, that blocks the pipe flow. A collider junction \underline{c} is like a blackhole or a huge leak. Its presence blocks passage of the fluid as long as neither \underline{c} nor any of the descendants of \underline{c} are in \underline{Z} . If, on the other hand, $\underline{c} \in \underline{Z}$, or $\underline{c}' \in \underline{Z}$ where $\underline{c}' \in de(\underline{c})$, then that acts as a complete (in the case of $\underline{c} \in \underline{Z}$) or a partial (in the case of $\underline{c}' \in \underline{Z}$) bridge across the blackhole.

See Fig.19.1 for some examples of paths that are blocked or not blocked by conditioning on a multinode \underline{Z} .

¹Note that we speak of blocked paths or info, not of blocked nodes. Nodes are not blocked; rather they are either conditioned upon or not.

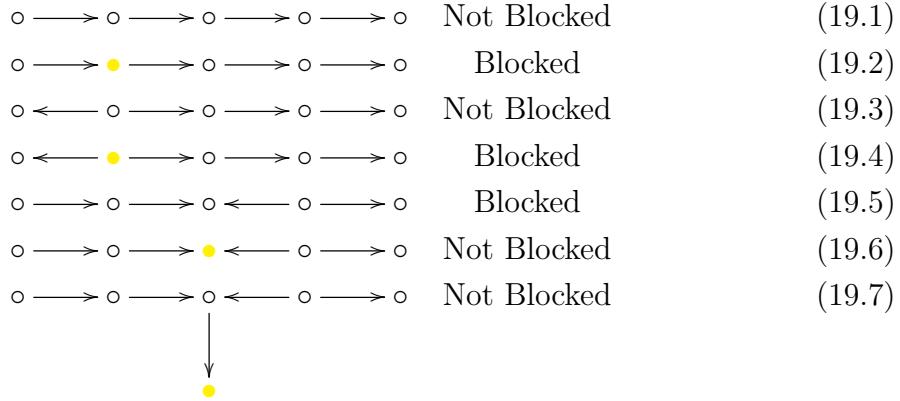


Figure 19.1: Examples of paths that are blocked or not blocked by conditioning on a multinode $\underline{Z}.$ Nodes belonging to $\underline{Z}.$ are colored yellow.

Given 3 disjoint multinode $\underline{A}., \underline{B}., \underline{Z}.$ of a graph G , we write “ $\underline{A}. \perp_G \underline{B}.|\underline{Z}.$ ” or say “ $\underline{A}.$ and $\underline{B}.$ are **d-separated** by $\underline{Z}.$ in G ” iff there exists no path γ from $\underline{a} \in \underline{A}.$ to $\underline{b} \in \underline{B}.$ which is not blocked by conditioning on $\underline{Z}..$ ²

The minimal Markov blanket (see Chapter 39) of a node \underline{a} is the smallest multinode $\underline{Z}.$ such that $\underline{a} \perp_G \underline{b}|\underline{Z}.$ for all $\underline{b} \notin \underline{a} \cup \underline{Z}..$

We are finally ready to state the d-separation theorem, without proof.

A probability distribution P is **compatible with a DAG** G if P and G have the same random variables, and they can be combined to form a bnet without contradictions; i.e., one can calculate all the TPMs from P and multiply them together to obtain P again.

Claim 35 (d-separation Theorem)

Suppose $\underline{A}., \underline{B}., \underline{Z}.$ are disjoint multinode of a DAG G .

If $\underline{A}. \perp_G \underline{B}.|\underline{Z}.$ then $P(\underline{B}.|\underline{A}.,\underline{Z}.) = P(\underline{B}.|\underline{Z}.)$ for all $\underline{B}., \underline{A}., \underline{Z}.$ for all P compatible with G .

The full converse of the theorem can also be proven, but we won't be using it in this book.

Often, the right hand side of this theorem is stated as “ $\underline{A}. \perp_P \underline{B}.|\underline{Z}.$ for all P ”. Then the theorem is stated: “If $\underline{A}. \perp_G \underline{B}.|\underline{Z}.$ then $\underline{A}. \perp_P \underline{B}.|\underline{Z}.$ for all P .”

Note that the following are equivalent:

- $P(\underline{B}.|\underline{A}.,\underline{Z}.) = P(\underline{B}.|\underline{Z}.)$ for all $\underline{B}., \underline{A}., \underline{Z}..$
- $\underline{A}. \perp_P \underline{B}.|\underline{Z}.$

² $\underline{Z}.$ are the nodes we are “conditioning on”. Unmeasured (i.e., hidden, unobserved) nodes cannot be conditioned on, because that would entail measuring them.

- $H(\underline{A}_. : \underline{B}_. | \underline{Z}_.) = 0$ (see Chapter Notational Conventions and Preliminaries for definition of conditional mutual information (CMI))

Extra stuff: mostly only for pure mathematicians

Below, we will use the notation $nde(\underline{a})$ to denote all non-descendants, including \underline{a} itself, of a node \underline{a} in a DAG G ; i.e., all nodes of G that are not in $de(\underline{a}) \cup \underline{a}$, where $de(\underline{a})$ is defined in Chapter Definition of a Bayesian Network.

Given a DAG G , define the following sets of d-separations:³

$$DS(G) = \{(\underline{A}_. \perp_G \underline{B}_. | \underline{Z}_.) : \underline{A}_., \underline{B}_., \underline{Z}_. \text{ are multinodes of } G\} . \quad (19.8)$$

$$DS_{min}(G) = \{(\underline{A}_. \perp_G nde(\underline{A}_.) | pa(\underline{A}_.)) : \underline{A}_. \text{ is a multinode of } G\} . \quad (19.9)$$

See Chapter 53 for an example where set $DS_{min}(G)$ is calculated for a particular DAG G .

Claim 36 *For all DAGs G , $DS(G) = DS_{min}(G)$.*

Given a probability distribution P , define the following set of conditional independencies:

$$CI(P) = \{(\underline{A}_. \perp_P \underline{B}_. | \underline{Z}_.) : \underline{A}_., \underline{B}_., \underline{Z}_. \text{ are multinodes of } P\} , \quad (19.10)$$

For a DAG G and a probability distribution P compatible with G , define a map ϕ by

$$\phi : DS_{min}(G) \rightarrow CI(P) \quad (19.11)$$

$$\phi : \underline{A}_. \perp_G nde(\underline{A}_.) | pa(\underline{A}_.) \mapsto \underline{A}_. \perp_P nde(\underline{A}_.) | pa(\underline{A}_.) \quad (19.12)$$

In general, this map is 1-1 but not onto.

Claim 37 *For a bnet with a DAG G and a total probability distribution P , the map ϕ is a bijection.*

$DS(G)$ does not fully specify a DAG. DAGs with the same $DS(G)$ are said to be **d-separation equivalent**. See Chapter 53 for more info about d-separation equivalence.

³Note that $(\underline{A}_. \perp_G nde(\underline{A}_.) | pa(\underline{A}_.))$ and $(\underline{A}_. \perp_G nde(\underline{A}_.) - pa(\underline{A}_.) | pa(\underline{A}_.))$ are equivalent because $H(\underline{a} : \underline{b}, \underline{c} | \underline{c}) = H(\underline{a} : \underline{b} | \underline{c})$.

Chapter 20

D-Separation and Do Calculus Rules for LDEN: COMING SOON

In this chapter, we will prove the D-Separation Theorem (see Chapter 19) and the Do Calculus Rules (see Chapter 17) for the special bnets that we call LDEN(see Chapter 38). Proving these theorems for LDEN rather than for general bnets is much simpler and quite instructive.

Chapter 21

D-Separation in Quantum Mechanics

See Ref.[61].

Chapter 22

Dynamical Bayesian Networks

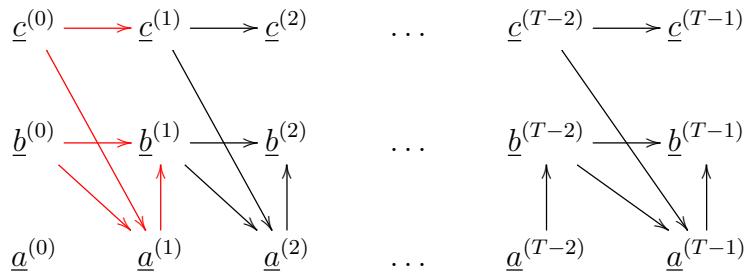


Figure 22.1: Example of a dynamical bnet. The pattern of red arrows is repeated $T - 1$ times.

A dynamical bnet is simply a time homogeneous Markov chain (see Chapter 41) for which each node is called a **time slice**, and each time slice represents at finer resolution a sub-DAG which is the same between any 2 successive time slices. Fig.22.1 gives an example of a dynamical bnet. In Fig.22.1, we've drawn the 3 nodes of each time slice vertically, and labeled them with a superscript $^{(t)}$, where $t \in \{0, 1, \dots, T-1\}$ is the time of the slice. To fully specify the dynamical bnet of Fig.22.1, we would also have to specify the TPMs

$$\begin{aligned} &P(c^{(0)}), \\ &P(b^{(0)}) \\ &P(c^{(1)}|c^{(0)}), \\ &P(b^{(1)}|b^{(0)}, a^{(1)}) \\ &P(a^{(1)}|b^{(0)}). \end{aligned}$$

Dynamical bnets are very common in AI and Data Science. Kalman filters (Chapter 36), Hidden Markov Models (Chapter 30) and Recurrent Neural Networks (Chapter 59) are famous examples of dynamical bnets.

Bnets are acyclic; they can't have cycles (i.e., closed directed paths). Yet feedback loops are an important concept in Science. So what is the equivalent of feedback loops in the bnet world? Dynamical bnets are. Fig.22.2 represents Fig.22.1

more compactly using feedback loops. Any bnet with feedback loops can be “unrolled” into a dynamical bnet.

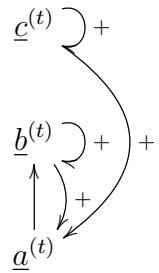


Figure 22.2: Dynamical bnet Fig.22.1 represented more compactly using feedback loops. Arrows labelled + point from nodes of the t time slice to nodes of the $t + 1$ time slice.

Chapter 23

Expectation Maximization

This chapter is based on Refs.[86] and [137].

The Expectation Maximization (EM) algorithm is commonly used in Data Science to find the maximum over an **unknown parameter** θ of a likelihood function

$$P(\vec{x}|\theta) = \sum_{\vec{h}} P(\vec{x}, \vec{h}|\theta), \quad (23.1)$$

where \vec{x} denotes the **observed variables**, and \vec{h} denotes the **hidden variables**. Both θ and \vec{h} are hidden (i.e., unobserved).¹

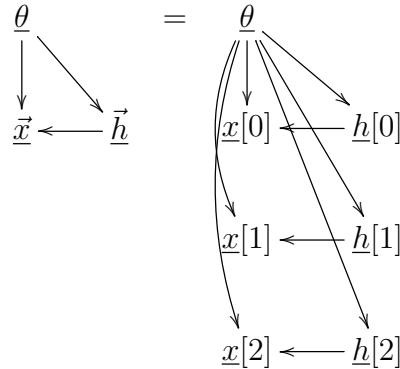


Figure 23.1: bnet for EM with $nsam = 3$. $x[\sigma] = x^\sigma$ and $h[\sigma] = h^\sigma$.

The bnet for the EM algorithm is given by Fig.23.1 for $nsam = 3$. Later on in this chapter, we will give the node TPMs for this bnet for the special case in which $P(x^\sigma | \theta)$ is a mixture (i.e., weighted sum) of Gaussians.

¹The term “unknown parameter” is mainly of frequentist origin. For Bayesians, θ is a random variable with a delta function prior, whereas for frequentists, it is not a random variable at all, just an unknown parameter with no randomness.

Note that if we erase the \underline{h}^σ nodes from Fig.23.1, we get the bnet for naive Bayes, which is used for classification into the states of $\underline{\theta}$. However, there is one big difference. With naive Bayes, the leaf nodes have different TPMs. Here, we will assume they are i.i.d. Naive Bayes is used for classification: i.e., given the states of the leaf nodes, we infer the state of the root node. EM is used for clustering; i.e., given many i.i.d. samples, we fit their distribution by a weighted sum of prob distributions, usually Gaussians.

Let

\mathcal{L} =likelihood function.

$nsam$ = number of samples.

$\vec{x} = (x[0], x[1], \dots, x[nsam - 1])$, $x^\sigma = x[\sigma] \in S_{\underline{x}}$ for all σ .

$\vec{h} = (h[0], h[1], \dots, h[nsam - 1])$, $h^\sigma = h[\sigma] \in S_{\underline{h}}$ for all σ .

We assume that the samples (x^σ, h^σ) are i.i.d. for different σ at fixed θ . What this means is that there are probability distributions $P_{\underline{x}|\underline{h},\underline{\theta}}$ and $P_{\underline{h}|\underline{\theta}}$ such that

$$P(\vec{x}, \vec{h} | \theta) = \prod_{\sigma} [P_{\underline{x}|\underline{h},\underline{\theta}}(x^\sigma | h^\sigma, \theta) P_{\underline{h}|\underline{\theta}}(h^\sigma | \theta)] . \quad (23.2)$$

Definition of likelihood functions:

$$\underbrace{P(\vec{x} | \theta)}_{\mathcal{L}(\theta; \vec{x})} = \sum_{\vec{h}} \underbrace{P(\vec{x}, \vec{h} | \theta)}_{\mathcal{L}(\theta; \vec{x}, \vec{h})} \quad (23.3)$$

θ^* = maximum likelihood estimate of θ (no prior $P(\theta)$ assumed):

$$\theta^* = \operatorname{argmax}_{\theta} \mathcal{L}(\theta; \vec{x}) \quad (23.4)$$

23.1 The EM algorithm:

1. Expectation step:²

$$Q(\theta | \theta^{(t)}) = E_{\vec{h}|\vec{x},\theta^{(t)}} \ln P(\vec{x}, \vec{h} | \theta) \quad (23.5)$$

2. Maximization step:

$$\theta^{(t+1)} = \operatorname{argmax}_{\theta} Q(\theta | \theta^{(t)}) . \quad (23.6)$$

Claim: $\lim_{t \rightarrow \infty} \theta^{(t)} = \theta^*$.

Fig.23.2 portrays the recursive nature of the EM algo as a dynamical, recurrent bnet. For Fig.23.2, the TPMs, printed in blue, for the $\underline{\theta}^{(t)}$ nodes for $t = 1, 2, \dots$, are as follows:

²Note that the right hand side of Eq.(23.5) is expressible in the form $\sum_{\sigma} \sum_{h^\sigma} f(x^\sigma, h^\sigma)$.

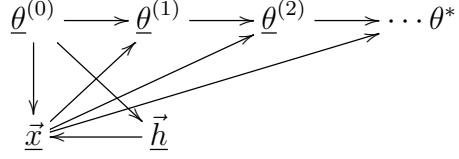


Figure 23.2: The EM algo generates a sequence of parameter estimates $(\theta^{(t)})_{t=0,1,2,\dots}$ that converges to the optimum (i.e., best-fit) parameter θ^* .

$$P(\theta^{(t+1)}|\vec{x}, \theta^{(t)}) = \delta(\theta^{(t+1)}, \operatorname{argmax}_{\theta} Q(\theta|\theta^{(t)})) . \quad (23.7)$$

23.1.1 Motivation

$$Q(\theta|\theta^{(t)}) = E_{\vec{h}|\vec{x}, \theta^{(t)}} \ln P(\vec{x}, \vec{h}|\theta) \quad (23.8)$$

$$= E_{\vec{h}|\vec{x}, \theta^{(t)}} [\ln P(\vec{h}|\vec{x}, \theta) + \ln P(\vec{x}|\theta)] \quad (23.9)$$

$$= -D_{KL} \left(P(\vec{h}|\vec{x}, \theta^{(t)}) \parallel P(\vec{h}|\vec{x}, \theta) \right) - H[P(\vec{h}|\vec{x}, \theta^{(t)})] + \ln P(\vec{x}|\theta) \quad (23.10)$$

When $\theta^{(t)} = \theta$, this becomes

$$Q(\theta|\theta) = -H[P(\vec{h}|\vec{x}, \theta)] + \ln P(\vec{x}|\theta) . \quad (23.11)$$

Hence,

$$\partial_{\theta} Q(\theta|\theta) = - \sum_{\vec{h}} \partial_{\theta} P(\vec{h}|\vec{x}, \theta) + \partial_{\theta} \ln P(\vec{x}|\theta) \quad (23.12)$$

$$= \partial_{\theta} \ln P(\vec{x}|\theta) \quad (23.13)$$

So if $\theta^{(t)} \rightarrow \theta$ and $Q(\theta|\theta)$ is max at $\theta = \theta^*$, then $\ln P(\vec{x}|\theta)$ is max at $\theta = \theta^*$ too.

For a more rigorous proof that $\lim_{t \rightarrow \infty} \theta^{(t)} = \theta^*$, see Wikipedia article Ref.[86] and references therein.

23.2 Minorize-Maximize (MM) algorithms

A function $\mu(\theta|\theta^{(t)})$ is said to **minorize** a target function $\mathcal{L}(\theta)$ iff for all θ at fixed $\theta^{(t)}$, it satisfies the “ $\mu \leq \mathcal{L}$ property”

$$\mu(\theta|\theta^{(t)}) \leq \mathcal{L}(\theta) , \quad (23.14)$$

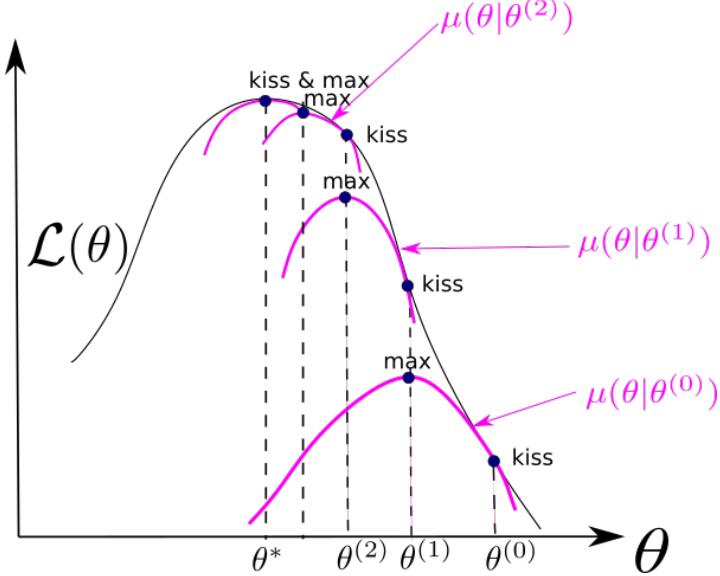


Figure 23.3: Function $\mu(\theta|\theta^{(t)})$ minorizes the function $L(\theta)$. Note that $\mu(\theta|\theta^{(t)})$ is always below $L(\theta)$. “max” indicates $\theta^{(t+1)} = \operatorname{argmax}_{\theta} \mu(\theta|\theta^{(t)})$. “kiss” indicates $\mu(\theta^{(t)}|\theta^{(t)}) = L(\theta^{(t)})$.

and the “ $\mu = L$ property”

$$\mu(\theta^{(t)}|\theta^{(t)}) = L(\theta^{(t)}) . \quad (23.15)$$

We **recursively maximize a minorizing function** $\mu(\theta|\theta^{(t)})$ if we define a sequence $(\theta^{(t)})_{t=0,1,\dots}$ as follows:

$$\theta^{(t+1)} = \operatorname{argmax}_{\theta} \mu(\theta|\theta^{(t)}) . \quad (23.16)$$

The sequence $(L(\theta^{(t)}))_{t=0,1,2,\dots}$ generated by recursively maximizing a minorizing function must be nondecreasing:

$$L(\theta^{(t+1)}) \geq \mu(\theta^{(t+1)}|\theta^{(t)}) \geq \mu(\theta^{(t)}|\theta^{(t)}) = L(\theta^{(t)}) . \quad (23.17)$$

A **minorize-maximize (MM) algorithm** is any algo that specifies a minorizing function $\mu(\theta|\theta^{(t)})$ for a particular target function $L(\theta)$. One can also define a **majorize-minimize algo (also called MM)** by inverting the inequalities throughout.

The EM algo is an MM algo. Indeed, if we define

$$L(\theta) = \ln P(\vec{x}|\theta) \quad (23.18)$$

and

$$\mu(\theta|\theta^{(t)}) = Q(\theta|\theta^{(t)}) + H(P(\vec{h}|\vec{x}, \theta^{(t)}), \quad (23.19)$$

then Eq.(23.10) establishes the $\mu \leq \mathcal{L}$ and $\mu = \mathcal{L}$ properties required of a minorizing function.

How an MM algo works is portrayed in Fig.23.3.

23.3 Examples

23.3.1 Gaussian mixture

$x^\sigma \in \mathbb{R}^d = S_{\underline{x}}$. $S_{\underline{h}}$ discrete and not too large. $n_{\underline{h}} = |S_{\underline{h}}|$ is number of Gaussians that we are going to fit the samples with.

Let

$$\theta = [w_h, \mu_h, \Sigma_h]_{h \in S_{\underline{h}}}, \quad (23.20)$$

where $[w_h]_{h \in S_{\underline{h}}}$ is a probability distribution of weights, and where $\mu_h \in \mathbb{R}^d$ and $\Sigma_h \in \mathbb{R}^{d \times d}$ are the mean value vector and covariance matrix of a d -dimensional Gaussian distribution.

The TPMs, printed in blue, for the bnet Fig.23.1, for the special case of a mixture of Gaussians, are as follows:

$$P(x^\sigma | h^\sigma | \theta) = \mathcal{N}_d(x^\sigma; \mu_{h^\sigma}, \Sigma_{h^\sigma}) \quad (23.21)$$

$$P(h^\sigma | \theta) = w_{h^\sigma} \quad (23.22)$$

Note that

$$P(x^\sigma | \theta) = \sum_h P(x^\sigma | h^\sigma = h, \theta) P(h^\sigma = h | \theta) \quad (23.23)$$

$$= \sum_h w_h \mathcal{N}_d(x^\sigma; \mu_h, \Sigma_h) \quad (23.24)$$

$$P(\vec{x}, \vec{h} | \theta) = \prod_\sigma [w_{h^\sigma} \mathcal{N}_d(x^\sigma; \mu_{h^\sigma}, \Sigma_{h^\sigma})] \quad (23.25)$$

$$= \prod_\sigma \prod_h [w_h \mathcal{N}_d(x^\sigma; \mu_h, \Sigma_h)]^{\mathbb{1}(h=h^\sigma)} \quad (23.26)$$

Old Faithful: See Wikipedia Ref.[86] for an animated gif of a classic example of using EM to fit samples with a Gaussian mixture. Unfortunately, could not include

it here because pdflatex does not support animated gifs. The gif shows samples in a 2 dimensional space (eruption time, delay time) from the Old Faithful geyser. In that example, $d = 2$ and $n_h = 2$. Two clusters of points in a plane are fitted by a mixture of 2 Gaussians.

K-means clustering is often presented as the main competitor to EM for doing **clustering (non-supervised learning)**. In K-means clustering, the sample points are split into K mutually disjoint sets S_0, S_1, \dots, S_{K-1} . The algorithm is easy to describe:

1. Initialize by choosing at random K data points $(\mu_k)_{k=0}^{K-1}$ called means or centroids and placing μ_k in S_k for all k .
2. **STEP 1:** For each data point, add it to the S_k whose centroid μ_k is closest to it.
3. **STEP 2:** Recalculate the centroids. Set μ_k equal to the mean value of set S_k .
4. Repeat steps 1 and 2 until the centroids stop changing by much.

Step 1 is analogous to the expectation step in EM, and Step 2 to the maximization step in EM (θ estimation versus μ_k estimation). We won't say anything further about K-means clustering because it isn't related to bnets in any way, and this is a book about bnets. For more info about K-means clustering, see Ref.[100].

23.3.2 Blood Genotypes and Phenotypes

Notation: $\vec{a} = (\underline{a}^\sigma)_{\sigma=0,1,\dots,n_{sam}-1}$, where n_{sam} is the number of samples.

Suppose $\vec{x} = (\vec{x}_0)$ (i.e., just one component)

$\vec{h} = (\vec{h}_0)$ (i.e., just one component)

$\underline{h}^\sigma \in S_h = \{AA, AO, BB, BO, OO, AB\}$ (the 6 blood genotypes)

$\underline{x}^\sigma \in S_x = \{A, B, O, AB\}$ (the 4 blood phenotypes)

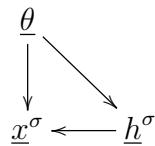


Figure 23.4: bnet for blood phenotypes x^σ and genotypes h^σ .

For the bnet Fig.23.4, the TPMs, printed in blue, are as follows:

$$P(h^\sigma | \theta) = \begin{array}{c|c} & \\ \hline AA & p_A^2 \\ AO & 2p_A p_O \\ BB & p_B^2 \\ BO & 2p_B p_O \\ OO & p_O^2 \\ AB & 2p_A p_B \end{array}, \quad (23.27)$$

where $p_A + p_B + p_O = 1$.

$$P(x^\sigma | h^\sigma, \theta) = \begin{array}{c|cccccc} & AA & AO & BB & BO & OO & AB \\ \hline A & 1 & 1 & 0 & 0 & 0 & 0 \\ B & 0 & 0 & 1 & 1 & 0 & 0 \\ O & 0 & 0 & 0 & 0 & 1 & 0 \\ AB & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \quad (23.28)$$

$$\theta = (p_A, p_B) \quad (23.29)$$

Multiplying the TPMs in Eqs.(23.27 and (23.28), we get

$$P(x^\sigma | \theta) = \begin{array}{c|c} & \\ \hline A & p_A^2 + 2p_A p_O (= \pi_A) \\ B & p_B^2 + 2p_B p_O (= \pi_B) \\ O & p_O^2 (= \pi_O) \\ AB & 2p_A p_B (= \pi_{AB}) \end{array} \quad (23.30)$$

Note that

$$P(\vec{x} | \theta) = \prod_{\sigma} P(x^\sigma | \theta) \quad (23.31)$$

$$= (\pi_A)^{N_A} (\pi_B)^{N_B} (\pi_O)^{N_O} (\pi_{AB})^{N_{AB}}, \quad (23.32)$$

where N_x for $x \in S_x = \{A, B, O, AB\}$ are the counts from the data. We can get estimates for the parameters p_A and p_B right here without doing EM. Just note that

$$\hat{\pi}_x = \frac{N_x}{N_+} \quad (23.33)$$

for $x \in S_x$, where $N_+ = \sum_x N_x$. Eqs.(23.33) give 4 quadratic equations that can be solved for the parameters p_A, p_B in terms of the observed counts N_x for $x \in S_x$.

If, instead, you want to find the optimum parameters p_A, p_B using EM, note that

$$Q(\theta|\theta^{(t)}) = \sum_{\vec{h}} P(\vec{h}|\theta^{(t)}) \ln P(\vec{x}, \vec{h}|\theta) \quad (23.34)$$

$$= \sum_{\vec{h}} \left[\prod_{\sigma} P(h^{\sigma}|\theta^{(t)}) \right] \ln \left[\prod_{\sigma} P(x^{\sigma}, h^{\sigma}|\theta) \right] \quad (23.35)$$

$$= \sum_{\sigma} \sum_{h^{\sigma}} P(h^{\sigma}|\theta^{(t)}) \ln P(x^{\sigma}, h^{\sigma}|\theta) \quad (23.36)$$

$$= \sum_{\sigma} \sum_{h^{\sigma}} P(h^{\sigma}|\theta^{(t)}) [\ln P(x^{\sigma}|h^{\sigma}, \theta) + \ln P(h^{\sigma}|\theta)] \quad (23.37)$$

$$= nsam \sum_{h^{\sigma}} P(h^{\sigma}|\theta^{(t)}) \ln P(h^{\sigma}|\theta). \quad (23.38)$$

23.3.3 Missing Data/Imputation

The previous example on blood genotypes and phenotypes assumed no missing data in compiling the counts N_x . But what if there is missing data? Can one still apply the EM algo in that case? Yes! See Chapter 46.

Chapter 24

Frisch-Waugh-Lovell (FWL) theorem

The Frisch-Waugh-Lovell (FWL) theorem (see Ref.[87]) (mnemonic: FoWL Theorem) is a method used in Linear Regression (LR). It allows us to calculate a regression coefficient by doing LR with two residuals. Let's call them the **independent residual** and the **dependent or target residual**.¹ These two residuals are calculated with the help of two previously performed LR steps.

As in the section Linear regression, Ordinary Least Squares (OLS) on LR, we will consider two cases: x^σ non-random, and x^σ random i.i.d..

24.1 FWL, assuming x^σ are non-random

Suppose

$$y = X_1\beta_1 + X_2\beta_2 + \epsilon \quad (24.1)$$

where

$$\begin{aligned} y, \epsilon &\in \mathbb{R}^{nsam} \\ X_a &\in \mathbb{R}^{nsam \times k_a}, \beta_a \in \mathbb{R}^{k_a} \text{ for } a = 1, 2 \\ \text{Define the matrices } U_1 \text{ and } A_1 \text{ by} \end{aligned}$$

$$U_1 = X_1(X_1^T X_1)^{-1} X_1^T \quad (24.2)$$

and

$$A_1 = I - U_1. \quad (24.3)$$

Note that

$$U_1 X_1 = X_1, \quad A_1 X_1 = 0 \quad (24.4)$$

¹Standard LR is performed with two features, the independent feature and the dependent or target feature.

(mnemonic: A_1 Annihilates X_1 , and U_1 acts like Unity on X_1).

Applying A_1 to Eq.(24.1) gives

$$A_1y = A_1X_2\beta_2 + A_1\epsilon \quad (24.5)$$

so we can estimate β_2 by

$$\boxed{\hat{\beta}_2 = (A_1X_2)^{-1}A_1y} \quad (24.6)$$

24.2 FWL, assuming x^σ are random

Assume for simplicity that $k_1 = k_2 = 1$ in Eq.(24.1). Let $\beta_1 = \alpha \in \mathbb{R}$, $\beta_2 = \beta \in \mathbb{R}$. When the x^σ are random and i.i.d., and X_1, X_2 are replaced by the random variables $\underline{x}, \underline{d} \in \mathbb{R}$, Eq.(24.1) is equivalent to

$$\underline{y} = \alpha\underline{x} + \beta\underline{d} + \underline{u}_y. \quad (24.7)$$

As usual, we will denote the linear operator $\langle \underline{x}, \cdot \rangle^{-1} \langle \underline{x}, \cdot \rangle$ by a derivative operator:

$$\frac{\langle \underline{x}, \cdot \rangle}{\langle \underline{x}, \underline{x} \rangle} = \frac{d \cdot}{d\underline{x}}. \quad (24.8)$$

Define operators $U_{\underline{x}}$ and $A_{\underline{x}}$ by

$$U_{\underline{x}}(\cdot) = \underline{x} \frac{d \cdot}{d\underline{x}} \quad (24.9)$$

and

$$A_{\underline{x}} = 1 - U_{\underline{x}}. \quad (24.10)$$

Note that

$$U_{\underline{x}}\underline{x} = \underline{x}, \quad A_{\underline{x}}\underline{x} = 0. \quad (24.11)$$

If we apply $A_{\underline{x}}$ to Eq.(24.7), we get

$$A_{\underline{x}}\underline{y} = \beta A_{\underline{x}}\underline{d} + A_{\underline{x}}\underline{u}_y. \quad (24.12)$$

Hence,

$$\langle A_{\underline{x}}\underline{d}, A_{\underline{x}}\underline{y} \rangle = \beta \langle A_{\underline{x}}\underline{d}, A_{\underline{x}}\underline{d} \rangle. \quad (24.13)$$

Thus,

$$\boxed{\beta = \frac{d(A_{\underline{x}}y)}{d(A_{\underline{x}}d)}}. \quad (24.14)$$

Eqs.(24.6) and (24.14) constitute our statement of the FWL theorem. They are the main result of this chapter. But before ending this chapter, let us show how the FWL theorem for random x^σ can be interpreted graphical using LDEN diagrams (LDEN diagrams are discussed in Chapter 38).

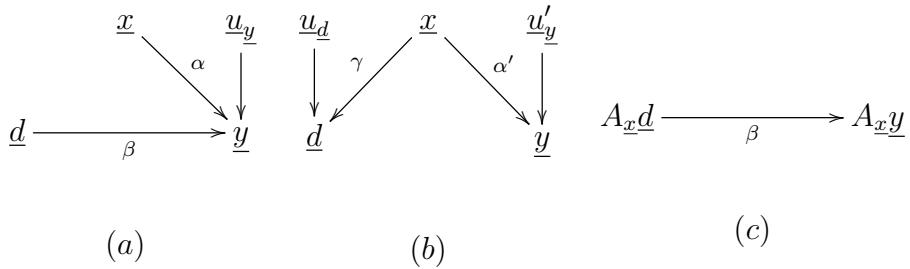


Figure 24.1: LDEN diagrams for discussing the FWL theorem.

Fig.24.1(a) is a graphical representation of the following equation:

$$y = \alpha \underline{x} + \beta \underline{d} + \underline{u}_y. \quad (24.15a)$$

Likewise, 24.1(b) represents

$$\begin{cases} \underline{d} = \gamma \underline{x} + \underline{u}_d \\ \underline{y} = \alpha' \underline{x} + \underline{u}'_y \end{cases} \quad (24.15b)$$

and 24.1(c) represents

$$A_{\underline{x}}y = \beta A_{\underline{x}}d. \quad (24.15c)$$

Applying $d/d\underline{x}$ to Eq.(24.15a) yields

$$\frac{dy}{d\underline{x}} = \alpha + \beta \frac{dd}{d\underline{x}}. \quad (24.16)$$

Applying $d/d\underline{x}$ to Eq.(24.15b) yields

$$\gamma = \frac{dd}{d\underline{x}}, \quad \alpha' = \frac{dy}{d\underline{x}}. \quad (24.17)$$

Therefore,

$$\alpha' = \alpha + \beta \gamma. \quad (24.18)$$

Applying $A_{\underline{x}}$ to Eq.(24.15b) yields

$$A_{\underline{x}}\underline{y} = \left(1 - \underline{x}\frac{d}{d\underline{x}}\right)\underline{y} = \underline{y} - \alpha'\underline{x} \quad (24.19)$$

and

$$A_{\underline{x}}\underline{d} = \left(1 - \underline{x}\frac{d}{d\underline{x}}\right)\underline{d} = \underline{d} - \gamma\underline{x}. \quad (24.20)$$

Therefore, Eq.(24.15c) is equivalent to

$$\underline{y} - \alpha'\underline{x} = \beta(\underline{d} - \gamma\underline{x}). \quad (24.21)$$

Chapter 25

Frontdoor Adjustment Formula

The frontdoor (FD) adjustment formula is proven in Chapter 17 from the rules of Do Calculus. The goal of this chapter is to give examples of the use of that theorem. We will restate the theorem in this chapter, sans proof. There is no need to understand the theorem's proof in order to use it. However, you will need to skim Chapter 17 in order to familiarize yourself with the notation used to state the theorem. This chapter also assumes that you are comfortable with the rules for checking for d-separation. Those rules are covered in Chapter 19.

Suppose that we have access to data that allows us to estimate a probability distribution $P(x., m., y.)$. Hence, the variables $\underline{x}., \underline{m}., \underline{y}.$ are ALL observed (i.e., not hidden). Then we say that the frontdoor $\underline{m}.$ satisfies the **frontdoor adjustment criterion** relative to $(\underline{x}., \underline{y}.)$ if

1. All directed paths from $\underline{x}.$ to $\underline{y}.$ are intercepted by (i.e., have a node in) $\underline{m}..$
2. All backdoor paths from $\underline{x}.$ to $\underline{m}.$ are blocked.
3. All backdoor paths from $\underline{m}.$ to $\underline{y}.$ are blocked by conditioning on $\underline{x}..$

Claim 38 *Frontdoor Adjustment Formula*

If $\underline{m}.$ satisfies the frontdoor criterion relative to $(\underline{x}., \underline{y}.)$, and $P(x., m.) > 0$, then

$$P(y.|D\underline{x}. = x.) = \sum_{m.} \underbrace{\left[\sum_{x'.} P(y.|x'., m.) P(x'.) \right]}_{P(y.|D\underline{m}. = m.)} \underbrace{P(m.|x.)}_{P(m.|D\underline{x}. = x.)} \quad (25.1)$$

$$= \sum_{x'.} \sum_{m.} \begin{array}{c} \diagdown \\ x. \longrightarrow \sum m. \longrightarrow y. \end{array} \quad (25.2)$$

where $\sum x'.$ and $\sum m.$ means nodes $\underline{x}'.$ and $\underline{m}.$ are summed over.

proof: See Chapter 17.

QED

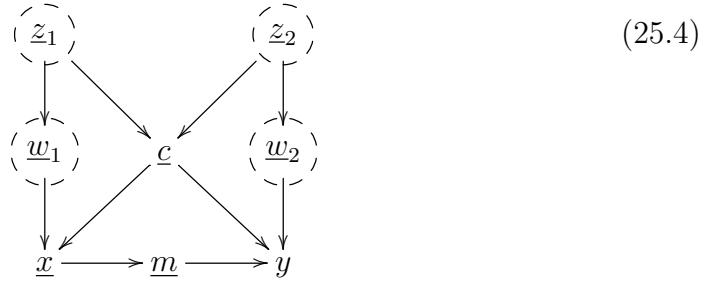
25.1 Examples

1.



If $x_+ = \underline{x}$, $m_+ = \underline{m}$ and $y_+ = \underline{y}$, then the FD criterion is satisfied. Can't satisfy backdoor criterion because \underline{c} unobserved so can't condition on it to block backdoor path $\underline{x} - \underline{c} - \underline{y}$.

2.



If $x_+ = \underline{x}$, $m_+ = \underline{m}$ and $y_+ = \underline{y}$, then the FD criterion is satisfied. Can't satisfy backdoor criterion because to block backdoor path $\underline{x} - \underline{c} - \underline{y}$, need to condition on \underline{c} but if this is true, then long path $\underline{x} - \underline{w}_1 - \underline{z}_1 - \underline{c} - \underline{z}_2 - \underline{w}_2 - \underline{y}$ becomes unblocked.

Chapter 26

Gaussian Nodes with Linear Dependence on Parents

Bnet nodes that have a Gaussian TPM with a linear dependence on their parent nodes (GLP) are a very popular way of modeling continuous nodes of bnets. A convenient aspect of them is that their parents can be discrete or continuous nodes, and their children can be discrete or continuous nodes too. Also, they can be learned easily from the data because their parameters can be expressed in terms of two node covariances. For these reasons, they are commonly used when doing structure learning of bnets with continuous nodes (see Chapter 70).

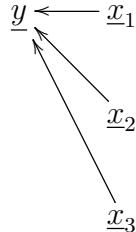


Figure 26.1: GLP node \underline{y} with 3 parent nodes $\underline{x}^3 = (\underline{x}_1, \underline{x}_2, \underline{x}_3)$.

Recall our notation for a Gaussian distribution:

$$\mathcal{N}(x; \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}, \quad (26.1)$$

where $x, \mu \in \mathbb{R}$ and $\sigma > 0$.

A GLP node \underline{y} with n parents $\underline{x}^n = (\underline{x}_1, \underline{x}_2, \dots, \underline{x}_n)$ has the following TPM:

$$P(\underline{y}|\underline{x}^n) = \mathcal{N}(\underline{y}; \beta_0 + \beta^{nT} \underline{x}^n, \sigma^2) \quad (26.2)$$

where $\underline{y}, \beta_0 \in \mathbb{R}$ and $\sigma^2 > 0$, and where $\underline{x}^n, \beta^n \in \mathbb{R}^n$ are **column vectors**. The T in β^{nT} stands for transpose. Any \underline{x}_i can have a discrete set of states as long as

they are real valued and ordinal (ordered by size). Fig.26.1 shows a diagrammatic representation of a GLP node with 3 parents.

Note that as $\sigma \rightarrow 0$, a GLP node becomes deterministic. In fact, it becomes a neural net node with a linear activation function.

An equivalent way of defining a GLP node \underline{y} is in terms of a random variable equation expressing \underline{y} as a hyperplane function of the parents \underline{x}^n plus a Gaussian noise variable. Define an estimator $\hat{\underline{y}}$ of a “true value” \underline{y} by

$$\hat{\underline{y}} = \beta_0 + \beta^{nT} \underline{x}^n \quad (26.3a)$$

and

$$\underline{y} = \hat{\underline{y}} + \underline{\epsilon} \quad (26.3b)$$

where the residual $\underline{\epsilon}$ satisfies

$$P(\epsilon) = \mathcal{N}(\epsilon; 0, \sigma^2) \quad (26.3c)$$

and

$$\langle \underline{x}^n, \underline{\epsilon} \rangle = 0 . \quad (26.3d)$$

The notation $\langle \underline{x}, \underline{y} \rangle$ for the covariance of random variables \underline{x} and \underline{y} is explained in Chapter Notational Conventions and Preliminaries.

Claim 39 *The parameters of a GLP node can be expressed in terms of 2-node covariances. Specifically,*

$$\beta^n = \langle \underline{x}^n, \underline{x}^{nT} \rangle^{-1} \langle \underline{y}, \underline{x}^n \rangle \quad (26.4)$$

$$\beta_0 = \langle \underline{y} \rangle - \beta^{nT} \langle \underline{x}^n \rangle \quad (26.5)$$

$$\sigma^2 = \langle \underline{y}, \underline{y} \rangle - \beta^{nT} \langle \underline{x}^n, \underline{y} \rangle \quad (26.6)$$

proof:

Note that $\langle \underline{x}^n, \underline{x}^{nT} \rangle^T = \langle \underline{x}^n, \underline{x}^{nT} \rangle$ and $\langle \underline{y}, \underline{x}^{nT} \rangle^T = \langle \underline{y}, \underline{x}^n \rangle$.

$$\langle \underline{y}, \underline{x}^{nT} \rangle = \beta^{nT} \langle \underline{x}^n, \underline{x}^{nT} \rangle \quad (26.7)$$

$$\langle \underline{y}, \underline{x}^n \rangle = \langle \underline{x}^n, \underline{x}^{nT} \rangle \beta^n \quad (26.8)$$

$$\beta^n = \langle \underline{x}^n, \underline{x}^{nT} \rangle^{-1} \langle \underline{y}, \underline{x}^n \rangle \quad (26.9)$$

$$\langle \underline{y} \rangle = \beta_0 + \beta^{nT} \langle \underline{x}^n \rangle \quad (26.10)$$

$$\langle \underline{y}, \underline{y} \rangle = \langle \beta_0 + \beta^{nT} \underline{x}^n + \underline{\epsilon}, \underline{y} \rangle \quad (26.11)$$

$$= \beta^{nT} \langle \underline{x}^n, \underline{y} \rangle + \sigma^2 \quad (26.12)$$

QED

Let D=Discrete, GLP=Gaussian with Linear dependence in Parents

The following arrows are possible in a bnet.

- $GLP \leftarrow GLP$

- $GLP \leftarrow D$

Pass to GLP a separate set of regression coefficients β_0, β^n and variance σ^2 for each state of D. If D is called \underline{d} , let

$$P(y|(x^n)_d, d) = \mathcal{N}(y; (\beta_0)_d + (\beta^{nT})_d(x^n)_d, \sigma_d^2) \quad (26.13)$$

for each $d \in S_{\underline{d}}$.

- $D \leftarrow GLP$

If D expects a continuous parent, no need to preprocess GLP output. If D expects a discrete parent, break the interval $[a, b]$ that contains most of the range of the GLP node into sub-intervals and assign a discrete label to each subinterval.

- $D \leftarrow D$

Chapter 27

Generative Adversarial Networks (GANs)

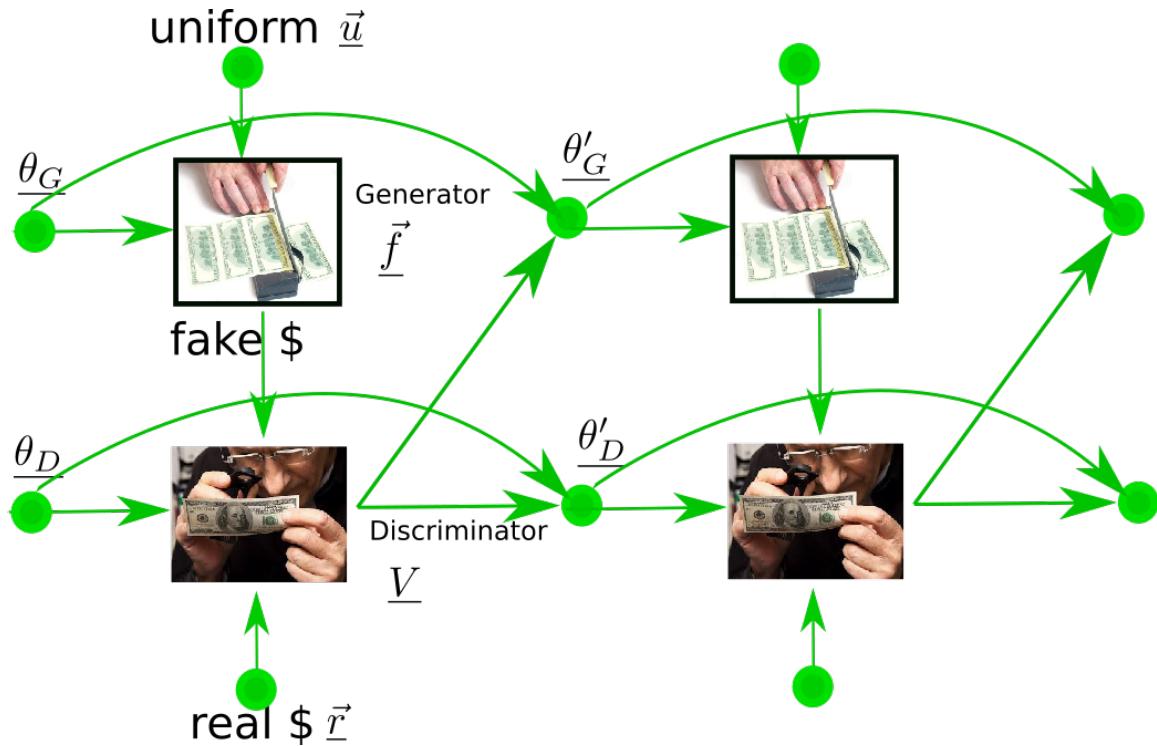


Figure 27.1: Generative Adversarial Network (GAN)

Original GAN, Ref.[12](2014).

Generator G (counterfeiter) generates samples \vec{f} of fake money and submits them to Discriminator D (Treasury agent). D also gets samples \vec{r} of real money. D submits verdict $V \in [0, 1]$. G depends on parameter θ_G and D on parameter θ_D . Verdict V and initial θ_G, θ_D are used to get new parameters θ'_G, θ'_D . Process is repeated.

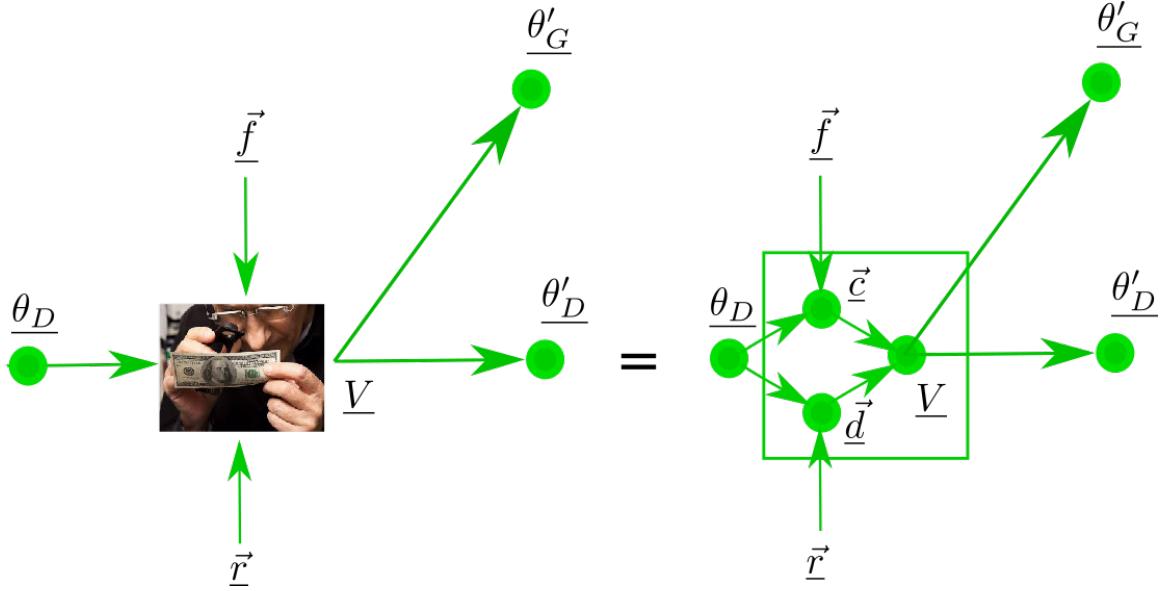


Figure 27.2: Discriminator node \underline{V} in Fig.27.1 can be split into 3 nodes \vec{c} , \vec{d} and \underline{V} .

(Dynamical Bayesian Network) until saddle point in $V(\theta_G, \theta_D)$ is reached. D makes G better and vice versa. Zero-sum game between D and G .

Let \mathcal{D} be the domain of $D(\cdot, \theta_D)$. Assume that for any $x \in \mathcal{D}$,

$$0 \leq D(x, \theta_D) \leq 1 . \quad (27.1)$$

For any $S \subset \mathcal{D}$, define

$$\sum_{x \in S} D(x, \theta_D) = \lambda(S, \theta_D) . \quad (27.2)$$

In general, $G(\cdot, \theta_G)$ need not be real valued.

Assume that for every $u \in S_u$, $G(u, \theta_G) = f \in S_f \subset \mathcal{D}$. Define

$$\overline{D}(f, \theta_D) = 1 - D(f, \theta_D) . \quad (27.3)$$

Note that

$$0 \leq \overline{D}(f, \theta_D) \leq 1 . \quad (27.4)$$

Define:

$$V(\theta_G, \theta_D) = \sum_r P(r) \ln D(r, \theta_D) + \sum_u P(u) \ln \overline{D}(G(u, \theta_G), \theta_D) . \quad (27.5)$$

We want the first variation of $V(\theta_G, \theta_D)$ to vanish.

$$\delta V(\theta_G, \theta_D) = 0 . \quad (27.6)$$

This implies

$$\partial_{\theta_G} V(\theta_G, \theta_D) = \partial_{\theta_D} V(\theta_G, \theta_D) = 0 \quad (27.7)$$

and

$$V_{opt} = \min_{\theta_G} \max_{\theta_D} V(\theta_G, \theta_D) . \quad (27.8)$$

The TPMs, printed in blue, for bnets Figs.27.1 and 27.2, are as follows:

$$P(\theta_G) = \text{ given} \quad (27.9)$$

$$P(\theta_D) = \text{ given} \quad (27.10)$$

$$P(\vec{u}) = \prod_i P(u[i]) \quad (\text{usually uniform distribution}) \quad (27.11)$$

$$P(\vec{r}) = \prod_i P(r[i]) \quad (27.12)$$

$$P(f[i] | \vec{u}, \theta_G) = \delta[f[i], G(u[i], \theta_G)] \quad (27.13)$$

$$P(c[i] | \vec{f}, \theta_D) = \delta(c[i], \overline{D}(f[i], \theta_D)) \quad (27.14)$$

$$P(d[j] | \vec{r}, \theta_D) = \delta(d[j], D(r[j], \theta_D)) \quad (27.15)$$

$$P(V|\vec{d}, \vec{c}) = \delta(V, \frac{1}{N} \ln \prod_{i,j} (c[i]d[j])) \quad (27.16)$$

where $N = nsam(\vec{r})nsam(\vec{u})$.

Let $\eta_G, \eta_D > 0$. Maximize V wrt θ_D , and minimize it wrt θ_G .

$$P(\theta'_G|V, \theta_G) = \delta(\theta'_G, \theta_G - \eta_G \partial_{\theta_G} V) \quad (27.17)$$

$$P(\theta'_D | V, \theta_D) = \delta(\theta'_D, \theta_D + \eta_D \partial_{\theta_D} V) \quad (27.18)$$

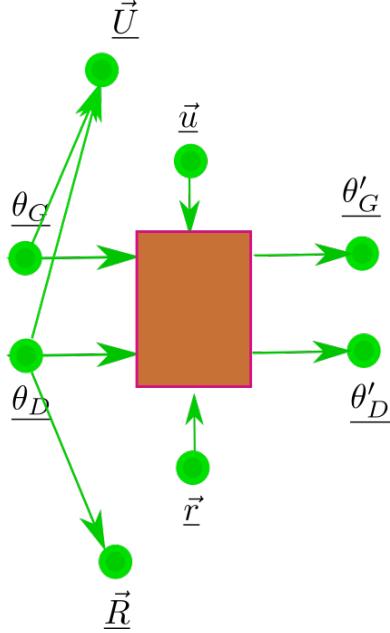


Figure 27.3: GAN, Constraining Bayesian Network

Constraining Bnet given in Fig.27.3. It adds 2 new nodes, namely \vec{U} and \vec{R} , to the bnet of Fig.27.1. The purpose of these 2 barren (childrenless) nodes is to constrain certain functions to be probability distributions.

The TPMs, printed in blue, for the 2 new nodes, are as follows.

$$P(U[i] | \theta_G) = \frac{\overline{D}(G(U[i], \theta_G), \theta_D)}{\bar{\lambda}(\theta_G, \theta_D)} \quad (27.19)$$

where $S_{\underline{U}[i]} = S_u$ and $\bar{\lambda}(\theta_G, \theta_D) = \sum_u \overline{D}(G(u, \theta_G), \theta_D)$.

$$P(R[i] | \theta_G, \theta_D) = \frac{D(R[i], \theta_D)}{\lambda(\theta_D)} \quad (27.20)$$

where $S_{\underline{R}[i]} = S_r$ and $\lambda(\theta_D) = \sum_r D(r, \theta_D)$.

$$P(V | \vec{u}, \vec{r}) = \delta(V, \frac{1}{N} \ln \prod_{i,j} (P(R[i] = r[i] | \theta_G, \theta_D) P(\underline{U}[i] = u[j] | \theta_G))) \quad (27.21)$$

where $N = nsam(\vec{r})nsam(\vec{u})$.

\mathcal{L} = likelihood

$$\mathcal{L} = P(\vec{r}, \vec{u} | \theta_G, \theta_D) \quad (27.22)$$

$$= \prod_{i,j} \left[\frac{D(r[i], \theta_D)}{\lambda(\theta_D)} \frac{\overline{D}(G(u[j], \theta_G), \theta_D))}{\bar{\lambda}(\theta_G, \theta_D)} \right] \quad (27.23)$$

$$\ln \mathcal{L} = N[V(\theta_G, \theta_D) - \ln \lambda(\theta_D) - \ln \bar{\lambda}(\theta_G, \theta_D)] \quad (27.24)$$

Chapter 28

Goodness of Causal Fit

This chapter is an almost verbatim copy of a paper of mine, Ref.[59].

28.1 Abstract

We propose a Goodness of Causal Fit (GCF) measure which depends on Pearl “do” interventions. This is different from a measure of Goodness of Statistical Fit (GF), which does not use interventions. Given a DAG set \mathcal{G} , to find a good $G \in \mathcal{G}$, we propose plotting $GCF(G)$ versus $GF(G)$ for all $G \in \mathcal{G}$, and finding a graph $G \in \mathcal{G}$ with a large amount of both types of goodness.

28.2 Introduction

Frequently, when students first encounter Bayesian Networks (bnets) and Causal Inference (CI), they experience serious doubts about the usefulness of this theory, because they believe finding the underlying causal model (i.e., DAG) for most realistic physical situations is too difficult or impossible. I think that part of the problem is that these students are assuming, perhaps unconsciously, that there exists a unique DAG that fits Nature perfectly, and a mind-boggling number of possibilities to sift through to find that DAG. Rather than looking for a unique DAG, I think a better strategy is to write down a set \mathcal{G} of likely DAGs, and to calculate for each DAG in \mathcal{G} , a measure called Goodness of Causal Fit (GCF). Then use the DAGs with the highest GCF scores.

The goal of this paper is to propose a GCF measure. Such a measure is of course not unique, and someone may propose in the future a measure that is better than ours.

When designing a GCF measure, it is important to keep in mind the First Dictum¹ of CI: The data is causal model-less. In the First Dictum, when we say “data”, we are referring to what is commonly called a dataset. A dataset is a table

¹This is just my whimsical name for it.

of data, where all the entries of each column have the same units, and measure a single feature, and each row refers to one particular sample or individual. Datasets are particularly useful for estimating probability distributions and for training neural nets. In the First Dictum, when we say “causal model”, we are referring to a DAG (directed acyclic graph) or a bnet (Bayesian Network= DAG + probability table for each node of DAG).

You can try to derive a causal model from a dataset, but you’ll soon find out that you can only go so far. The process of finding a *partial* causal model from a dataset is called structure learning (SL). SL can be done quite nicely with Marco Scutari’s open source program `bnlearn` (Ref[49]). The problem is that SL often cannot narrow down the causal model to a single one. It finds an undirected graph (UG), and it can determine the direction of some of the arrows in the UG, but it is often incapable, for well understood fundamental—not just technical—reasons, of finding the direction of *all* the arrows of the UG. So it often fails to fully specify a DAG model.

Let’s call the ordered pair (dataset, causal model) a **dataset++**. Then what the First Dictum is saying is that a dataset is causal model-free or causal model-less (although sometimes one can find a partial causal model hidden in there). A dataset is not a dataset++.

Graphs which contain both directed and undirected edges are called **partially directed (PD) graphs**. `bnlearn` takes a dataset as input and returns a PD graph G_{pd} . Given a PD graph G_{pd} , let $\mathcal{G}(G_{pd})$ be the DAG set \mathcal{G} which is generated by giving directions to all undirected edges of G_{pd} in all possible ways. We will refer to $\mathcal{G}(G_{pd})$ as the **DAG set generated by G_{pd}** . and to any $\mathcal{G}' \subset \mathcal{G}(G_{pd})$ as a **DAG set partially generated by G_{pd}** . Once we define below our GCF measure, we will apply it to sets of the type $\mathcal{G}(G_{pd})$ as an example.

It’s clear that any measure of GCF will have to involve interventions such as the “do” intervention invented by Pearl et al for CI. Without interventions like “do”, it is impossible to distinguish causally the DAGs in a set $\mathcal{G}(G_{pd})$.

Henceforth, random variables will be indicated by underlining.

28.3 Goodness of Statistical Fit

Before trying to define a GCF measure, it is instructive to review the closely related, well established, measures of Goodness of Statistical Fit (GF).

Consider two probability distributions $PO(x)$ and $PE(x)$, where $x \in S_{\underline{x}}$. By a GF measure, we mean a measure of the difference between PO and PE . Usually PO is the observed probability distribution and PE is the expected, theoretical one.

Three popular measures of the difference between PO and PE are:

1. The **Kullback-Liebler divergence**:

$$D_{KL}(PO \parallel PE) = \sum_{x \in S_{\underline{x}}} PO(x) \ln \frac{PO(x)}{PE(x)}. \quad (28.1)$$

2. The **Pearson divergence** (aka **Pearson Chi-squared test statistic**):

$$D_{\chi^2}(PO \parallel PE) = \sum_{x \in S_{\underline{x}}} \frac{[PO(x) - PE(x)]^2}{PE(x)} = \sum_{x \in S_{\underline{x}}} \frac{PO^2(x)}{PE(x)} - 1. \quad (28.2)$$

It's easy to show using $\ln(1 + \delta) = \delta + \mathcal{O}(\delta^2)$ that if $\left| \frac{PO(x)}{PE(x)} - 1 \right| \ll 1$ for all $x \in S_{\underline{x}}$, then

$$D_{KL}(PO \parallel PE) \approx D_{\chi^2}(PO \parallel PE) \quad (28.3)$$

3. The **Euclidean distance squared**:

$$D_E(PO, PE) = \sum_{x \in S_{\underline{x}}} [PO(x) - PE(x)]^2 \quad (28.4)$$

Note that of these 3 measures, only $D_E(PO, PE)$ is symmetric in PO and PE .

Given any bnet G with full probability distribution² $P_G(x.)$ and a probability distribution³ $\tilde{P}(x.)$ derived empirically from a dataset, let

$$D(G) = \sum_{x.} \tilde{P}(x.) \ln \frac{\tilde{P}(x.)}{P_G(x.)} \quad (28.5)$$

$$= D_{KL}(\tilde{P}(\underline{x.}) \parallel P_G(\underline{x.})) \quad (28.6)$$

We define **Goodness of Statistical Fit (GF)** of the bnet G by

$$GF(G) = \ln \frac{1}{D(G)} \quad (28.7)$$

28.4 GCF example 1

For the first example of our measure of GCF, we consider $\mathcal{G}(G_{pd}) = \{G_1, G_2\}$ given by Fig.28.1. We will assume the following:

²We define $x.$ to be a vector with components x_i

³Empirical distributions will be denoted by P with a tilde over it.

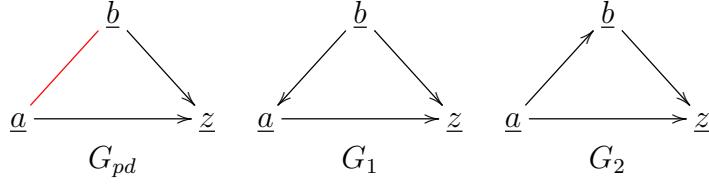


Figure 28.1: $\mathcal{G}(G_{pd}) = \{G_1, G_2\}$. The partially directed graph G_{pd} generates the DAGs G_1 and G_2 by giving directions to all undirected edges of G_{pd} in all possible ways. (In this case, there is only one undirected edge in G_{pd} .)

- First, we assume that we have collected a dataset from which we have extracted a full empirical distribution $\tilde{P}(z, a, b)$. From $\tilde{P}(z, a, b)$, we assume that the following have been calculated. $\tilde{P}(z, b|a)$ $\tilde{P}(z, a|b)$ $\tilde{P}(a)$, $\tilde{P}(b)$.
- Second, we assume that a dataset has been collected for which \underline{a} was held fixed to each of the possible values $a \in S_{\underline{a}}$ of \underline{a} . Furthermore, we assume that the distribution $\tilde{P}(z, b|do(\underline{a}) = a)$ has been extracted from that dataset.
- Third, we assume that a dataset has been collected for which \underline{b} was held fixed to each of the possible values $b \in S_{\underline{b}}$ of \underline{b} . Furthermore, we assume that the distribution $\tilde{P}(z, a|do(\underline{b}) = b)$ has been extracted from that dataset.

We will refer to $\tilde{P}(z, b|do(\underline{a}) = a)$ and $\tilde{P}(z, a|do(\underline{b}) = b)$ as **empirical do-probability distributions**.

Now define

$$D_a = \sum_{z,b} \tilde{P}(z, b|a) \ln \frac{\tilde{P}(z, b|a)}{\tilde{P}(z, b|do(\underline{a}) = a)} \quad (28.8)$$

$$= D_{KL}(\tilde{P}(z, b|a) \parallel \tilde{P}(z, b|do(\underline{a}) = a)) \quad (28.9)$$

$$D_{\underline{a}} = \sum_a \tilde{P}(a) D_a = E_a[D_a] \quad (28.10)$$

and

$$D_b = D_{KL}(\tilde{P}(z, \underline{a}|b) \parallel \tilde{P}(z, \underline{a}|do(\underline{b}) = b)) \quad (28.11)$$

$$D_{\underline{b}} = \sum_a \tilde{P}(b) D_b = E_b[D_b]. \quad (28.12)$$

We will refer to $D_{\underline{a}}$ and $D_{\underline{b}}$ as **do-divergences**.

Note that

$$D_a(G_2) = 0 \text{ for all } a \text{ so } \underbrace{D_{\underline{a}}(G_2)}_0 \leq D_{\underline{b}}(G_2) \quad (28.13)$$

and

$$D_b(G_1) = 0 \text{ for all } b \text{ so } D_{\underline{a}}(G_1) \geq \underbrace{D_b(G_1)}_0 . \quad (28.14)$$

If $D_{\underline{a}} \leq D_b$, then $\underline{a} \rightarrow b$, and if $D_{\underline{a}} \geq D_b$ then $\underline{a} \leftarrow b$. Thus, the arrow and the inequality sign point in opposite directions. Alternatively, just remember that the arrow points to the larger of the two D 's.

If $D_{\underline{a}} \leq D_b$, then define $GCF(G_1) = -1$ and $GCF(G_2) = 1$.

If $D_b \leq D_{\underline{a}}$, then define $GCF(G_1) = 1$, $GCF(G_2) = -1$.

28.5 GCF example 2

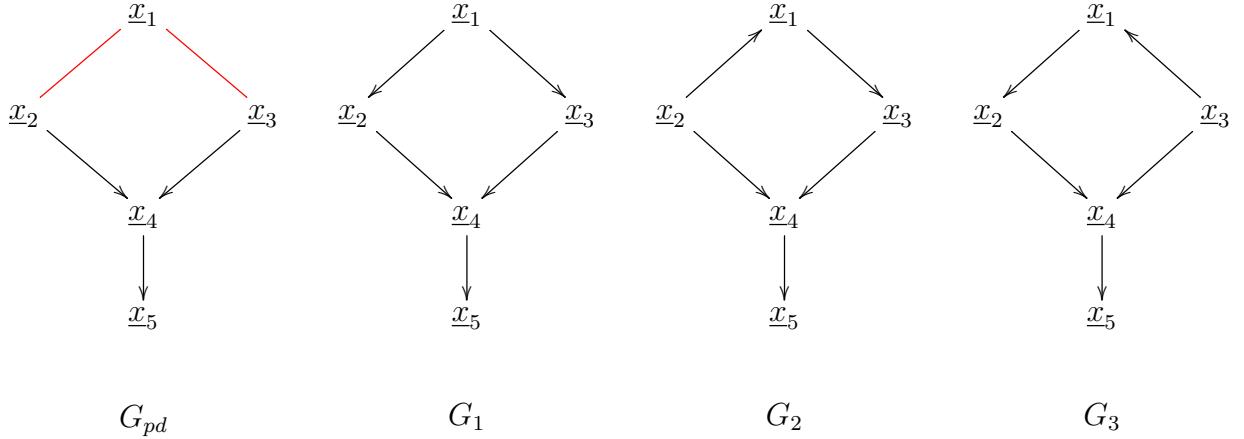


Figure 28.2: $\mathcal{G} = \{G_1, G_2, G_3\}$. \mathcal{G} is a set of observationally equivalent (OE) graphs. These are graphs that have the same value for GF, and are therefore indistinguishable from GF alone. For more info about OE graphs, see Chapter 53. Note that $\mathcal{G}(G_{pd})$ includes one more DAG, the one in which node \underline{x}_1 is a collider.

For the second example of our measure of GCF, consider $\mathcal{G} = \{G_1, G_2, G_3\}$ given by Fig.28.2.

Which of the do-divergences D_{x_2} , D_{x_1} and $D_{\underline{x}_3}$, is smallest, middle and highest, depends on the empirical do-probability distributions. For definiteness, suppose the sizes of these do-divergences are related as follows:

$$D_{\underline{x}_2} \leq D_{x_1} \leq D_{\underline{x}_3} . \quad (28.15)$$

For any two do-divergences $D_{\underline{a}}$ and D_b , define the distance

$$d_{b,\underline{a}} = |D_b - D_{\underline{a}}| \quad (28.16)$$

If we abbreviate $D_{\underline{x}_j}$ by D_j , we can define the GCF for each of the graphs in \mathcal{G} by:

$$GCF(G_1) = \frac{-d_{2,1} + d_{1,3}}{d_{2,1} + d_{1,3}} \quad (28.17a)$$

$$GCF(G_2) = \frac{d_{2,1} + d_{1,3}}{d_{2,1} + d_{1,3}} = 1 \quad (28.17b)$$

$$GCF(G_3) = \frac{-d_{2,1} - d_{1,3}}{d_{2,1} + d_{1,3}} = -1 \quad (28.17c)$$

28.6 GCF in general

Eqs.(28.17) are a special case of the following formulae.

For any two do-divergences $D_{\underline{a}}$ and $D_{\underline{b}}$, define the **do-divergence distance** by

$$d_{\underline{b}, \underline{a}} = |D_{\underline{b}} - D_{\underline{a}}| . \quad (28.18)$$

For any $G_i \in \mathcal{G}$, define the **edge-sign function** by

$$\sigma_{G_i}(\underline{a} - \underline{b}) = \begin{cases} +1 & \text{if edge } \underline{a} - \underline{b} \text{ in } G_i \text{ points towards larger of } D_{\underline{a}} \text{ and } D_{\underline{b}}. \\ -1 & \text{otherwise} \end{cases} \quad (28.19)$$

Finally, suppose that \mathcal{G} is either partially or fully generated by a PD graph G_{pd} with undirected edges $\{\underline{a}_k - \underline{b}_k\}_{k=0,1,\dots,n_k-1}$. Then define the GCF of graph $G_i \in \mathcal{G}$ by

$$GCF(G_i) = \frac{\sum_{k=0}^{n_k-1} \sigma_{G_i}(\underline{a}_k - \underline{b}_k) d_{\underline{a}_k, \underline{b}_k}}{\sum_{k=0}^{n_k-1} d_{\underline{a}_k, \underline{b}_k}} . \quad (28.20)$$

Note that $-1 \leq GCF(G_i) \leq 1$.

If the DAG set \mathcal{G} contains only one DAG G , define $GCF(G) = 1$, because all arrows in G are in the correct direction, as far as we know.

Call the **skeleton** of a DAG, the undirected graph that one obtains by turning all its edges from directed to undirected ones.

So far, we have applied our measure of GCF to a DAG set \mathcal{G} which is either fully or partially generated by a PD graph G_{pd} , or is a singleton set. But what if we want a GCF that can score every DAG in a DAG set \mathcal{G} that contains DAGs with different skeletons? In that case, let $\{\underline{a}_k - \underline{b}_k\}_{k=0,1,\dots,n_k-1}$ be *all* the edges of graph $G_i \in \mathcal{G}$, and use the relative GCF given by Eq.(28.20), or use an absolute GCF defined by

$$GCF_a(G_i) = \sum_{k=0}^{n_k-1} \sigma_{G_i}(\underline{a}_k - \underline{b}_k) d_{\underline{a}_k, \underline{b}_k} . \quad (28.21)$$

So let \mathcal{G} be an arbitrary DAG set. Our GCF (or GCF_a) measure is not enough to decide the best possible G in \mathcal{G} , because there might be several graphs with $GCF \approx 1$. For this reason, we recommend plotting $GCF(G)$ (or $GCF_a(G)$) versus $GF(G)$ for all $G \in \mathcal{G}$. Then choose a G with a large amount of both types of goodness. It might even be advantageous to average over a small subset of DAGs in \mathcal{G} that have large amounts of both types of goodness. This would be similar to averaging over an ensemble of decision trees to get a random forest.

A plot of $GCF(G)$ versus $GF(G)$ agrees with the spirit of the First Dictum and datasets++, because also in those, we acknowledge a separation between the dataset (GF) and causal model (GCF) degrees of freedom.

Chapter 29

Granger Causality

This chapter is based on the Wikipedia article Ref.[92] and Scholarpedia article Ref.[48] on Granger causality and on the book Ref.[14] on time series analysis by Hamilton.

This chapter assumes the reader has read Chapter 73 on the stationary time series $ARMA(p, q)$ and $VAR(p)$.

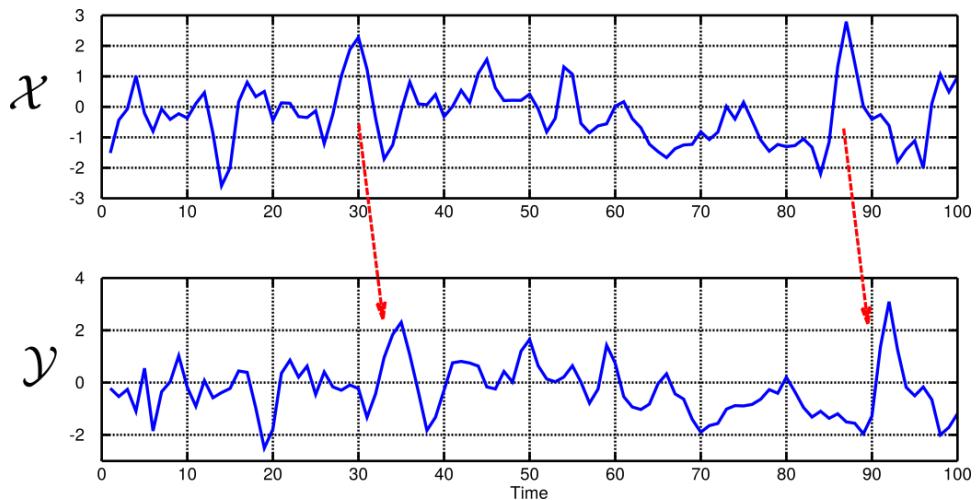


Figure 29.1: When t-series \mathcal{X} Granger-causes t-series \mathcal{Y} , the patterns in \mathcal{X} are approximately repeated in \mathcal{Y} after some time lag (two examples are indicated with arrows). Thus, past values of \mathcal{X} can be used for the prediction of future values of \mathcal{Y} . (image and caption from Ref.[92])

Let $\vec{x}_t = [x_t^A]_{\forall A} \in \mathbb{R}^{nr_1 \times 1}$ and $\vec{y}_t = [y_t^B]_{\forall B} \in \mathbb{R}^{nr_2 \times 1}$. Thus, \vec{x}_t for each t is a column vector with nr_1 rows, and \vec{y}_t for each t is a column vector with nr_2 rows. Let $nr_1 + nr_2 = nr$, the total number of rows. Consider a vector t-series $\{\vec{x}_t, \vec{y}_t\}_{\forall t}$ of type $VAR(p)$, as defined by Eq.(73.183). To simplify the notation of Eq.(73.183), we are replacing x^1 by x , x^2 by y , n^1 by n , and n^2 by w .

$$\begin{bmatrix} \underline{x}_t^A \\ \underline{y}_t^B \end{bmatrix} = \sum_{j=1}^p \begin{bmatrix} \alpha_j^{x|x;A,A'} & \alpha_j^{x|y;A,B'} \\ \alpha_j^{y|x;B,A'} & \alpha_j^{y|y;B,B'} \end{bmatrix} \begin{bmatrix} \underline{x}_{t-j}^{A'} \\ \underline{y}_{t-j}^{B'} \end{bmatrix} + \begin{bmatrix} \underline{n}_t^A \\ \underline{w}_t^B \end{bmatrix} \quad (29.1)$$

Hence,

$$E_{|\vec{x}_{<t}, \vec{y}_{<t}} \begin{bmatrix} \underline{y}_t^B \end{bmatrix} = \sum_{j=1}^p \alpha_j^{y|x;B,A'} \underline{x}_{t-j}^{A'} + \sum_{j=1}^p \alpha_j^{y|y;B,B'} \underline{y}_{t-j}^{B'} \quad (29.2)$$

Let $\mathcal{X} = \{\vec{x}_t\}_{\forall t}$ and $\mathcal{Y} = \{\vec{y}_t\}_{\forall t}$.

We say \mathcal{X} **does not G-cause** (or **does not G-forecast**) \mathcal{Y} , and symbolize this by $\mathcal{X} \dashv_G \mathcal{Y}$, if

$$\alpha_j^{y|x;B,A'} = 0 \quad \forall (j, B, A') \quad (29.3)$$

or, equivalently,

$$E_{|\vec{x}_{<t}, \vec{y}_{<t}} \begin{bmatrix} \underline{y}_t^B \end{bmatrix} = E_{|\vec{y}_{<t}} \begin{bmatrix} \underline{y}_t^B \end{bmatrix} \quad \forall (B, t) \quad (29.4)$$

We say \mathcal{X} **G-causes** (or **G-forecasts**) \mathcal{Y} and symbolize this by $\mathcal{X} \xrightarrow[G]{} \mathcal{Y}$, if $\mathcal{X} \dashv_G \mathcal{Y}$ is false.

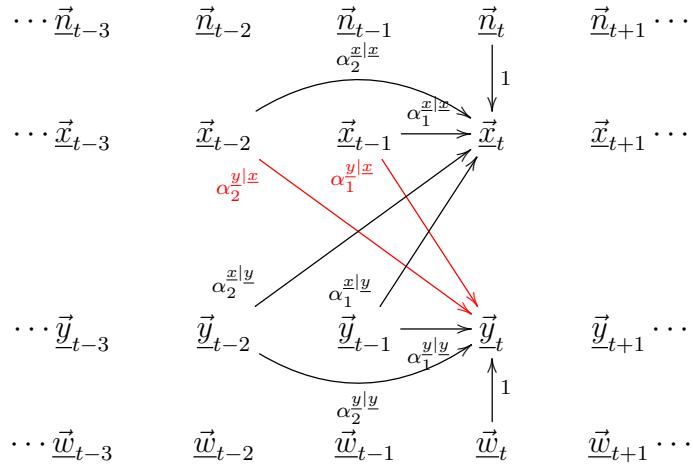


Figure 29.2: Bnet for $VAR(2)$. For clarity, we only show the arrows entering nodes \vec{x}_t and \vec{y}_t . Remove red arrows if \mathcal{X} does not G-cause \mathcal{Y} , and keep them if it does.

Eq.(29) describing a bipartite $VAR(p)$ t-series can be represented by the bnet Fig.29.2. The TPMs, printed in blue, for the two nodes \vec{x}_t and \vec{y}_t , are as follows:

$$P(\vec{x}_t | \vec{x}_{[t-p, t-1]}, \vec{y}_{[t-p, t-1]}, \vec{n}_t) = \prod_A \mathbb{1} \left(\underline{x}_t^A = \sum_{j=1}^p \alpha_j^{x|x;A,A'} \underline{x}_{t-j}^{A'} + \sum_{j=1}^p \alpha_j^{x|y;A,B'} \underline{y}_{t-j}^{B'} + n_t^A \right) \quad (29.5)$$

$$P(\vec{y}_t | \vec{x}_{[t-p,t-1]}, \vec{y}_{[t-p,t-1]}, \vec{w}_t) = \prod_B \mathbb{1} \left(\underline{y}_t^B = \sum_{j=1}^p \underbrace{\alpha_j^{y|x;B,A'}}_{0?} \vec{x}_{t-j}^{A'} + \sum_{j=1}^p \alpha_j^{y|y;B,B'} \vec{y}_{t-j}^{B'} + w_t^B \right) \quad (29.6)$$

Testing for GC

- Consider the datasets

$$\mathcal{D}_{\underline{x}} = \{(t, \vec{x}_{[t-p,t-1]}, \vec{y}_{[t-p,t-1]}, \boxed{\vec{x}_t}) : t\} \quad (29.7)$$

$$\mathcal{D}_{\underline{y}} = \{(t, \vec{x}_{[t-p,t-1]}, \vec{y}_{[t-p,t-1]}, \boxed{\vec{y}_t}) : t\} \quad (29.8)$$

One can do Linear Regression on $\mathcal{D}_{\underline{x}}$ with x-variables $(\vec{x}_{[t-p,t-1]}, \vec{y}_{[t-p,t-1]})$, y-variable \vec{x}_t , and regression coefficients $\alpha_{[1,p]}^{x|x}$, $\alpha_{[1,p]}^{x|y}$. One can also do Linear Regression on $\mathcal{D}_{\underline{y}}$ with x-variables $(\vec{x}_{[t-p,t-1]}, \vec{y}_{[t-p,t-1]})$, y-variable \vec{y}_t , and regression coefficients $\alpha_{[1,p]}^{y|x}$, $\alpha_{[1,p]}^{y|y}$. The LR software yields confidence intervals for the regression coefficients.

- One can do hypothesis testing using the Likelihood Ratio Test¹
Null hypothesis $H_0 : \mathcal{X} \xrightarrow[G]{\parallel} \mathcal{Y}$, Alternative hypothesis $H_1 : \mathcal{X} \xrightarrow[G]{} \mathcal{Y}$
- Test for both $\mathcal{X} \xrightarrow[G]{} \mathcal{Y}$ and $\mathcal{Y} \xrightarrow[G]{} \mathcal{X}$.

Limitations

It has been remarked that G-causality is not true causality because, even though an event A must precede an event B in order to cause it, that does not necessarily mean that A causes B. I think the problem with G-causality is that it uses a bnet that is a good fit for the dataset, but not necessarily also a good *causal* fit for the experiment. One can measure the goodness of causal fit of a bnet by doing do-intervention experiments (See Chapter 28).

¹The Likelihood Ratio Test is discussed in Section Maximum Likelihood Estimate, Likelihood Ratio Test.

Chapter 30

Hidden Markov Model

A dynamical Bayesian network (DBN) (see Chapter 22) is a generalization of a Hidden Markov Model (HMM), which in turn is a generalization of a Kalman Filter (KF) (see Chapter 36).

See Wikipedia article Ref.[93] to learn about the history and many uses of HMMs. This chapter is based on Refs.[32], [93], [135], [68].

In this chapter, we use the following conventions.

Random variables are underlined and their values are not. For example, $\underline{a} = a$ means the random variable \underline{a} takes the value a . Diagrams with nodes that are underlined represent Bayesian Networks (bnets) and the same diagram with the letters not underlined represents a specific **instantiation** of that bnet. For example $\underline{a} \rightarrow \underline{b}$ represents the bnet with conditional probability distribution $P(b|a)$, whereas $a \rightarrow b$ represents $P(b|a)$ itself.

If \underline{a} is a root node, then $\sum \underline{a}$ signifies a weighted sum $\sum_a P(a)$. For example, $\sum \underline{a} \rightarrow \underline{b} = \sum_a P(a)P(\underline{b}|a)$. If \underline{a} is not a root node as in $x \rightarrow \sum \underline{a} \rightarrow y = \sum_a P(y|a)P(a|x)$, then $\sum \underline{a}$ signifies a simple unweighted sum \sum_a .

Unobserved nodes are indicated by enclosing them in a dashed circle. For example, $\langle \underline{u} \rangle$.

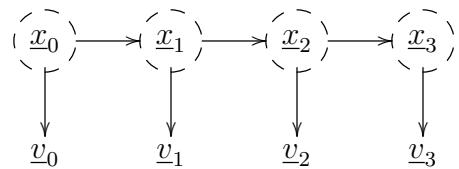


Figure 30.1: HMM bnet with $n = 4$.

Suppose

$v^n = (\underline{v}_0, \underline{v}_1, \dots, \underline{v}_{n-1})$ are n visible nodes that are measured, and
 $\underline{x}^n = (\underline{x}_0, \underline{x}_1, \dots, \underline{x}_{n-1})$ are the n hidden, unmeasurable state nodes of a system
that is being monitored.

For the bnet of Fig.30.1, one has

$$P(x^n, v^n) = \prod_{t=0}^{n-1} P(x_t|x_{t-1})P(v_t|x_t), \quad (30.1)$$

where $x_{-1} = \emptyset$.

We assume that this bnet is stationary. The following notation emphasizes that fact:

$\pi(x)$ = prior probability

$$\pi(x) = P(\underline{x}_0 = x) \quad (30.2)$$

$A(x|x')$ = transition matrix

$$A(x|x') = P(\underline{x}_t = x | \underline{x}_{t-1} = x') \quad (30.3)$$

$B(v|x)$ = emission probability

$$B(v|x) = P(\underline{v}_t = v | \underline{x}_t = x) \quad (30.4)$$

Let $x_{<t} = (x_0, x_1, \dots, x_{t-1})$.

For $t = 0, 1, \dots, n-1$, define

$\mathcal{F}_t(x_t)$ =future measurements probability

$$\mathcal{F}_t(x_t) = P(v_{>t}|x_t) \quad (30.5)$$

$$= \underline{x}_t \longrightarrow \sum x_{>t} \quad (30.6)$$

↓
 $v_{>t}$

$\overline{\mathcal{F}}_t(x_t)$ = past and present measurements probability

$$\overline{\mathcal{F}}_t(x_t) = P(v_{<t}, v_t, x_t) \quad (30.7)$$

$$= \sum x_{<t} \longrightarrow x_t \quad (30.8)$$

↓
 $v_{<t}$ ↓
 v_t

$\lambda_t(x_t)$ = present measurement probability, (a.k.a. emission probability $B(v_t|x_t)$). $\lambda_t(x_t)$ is the likelihood of x_t .

$$\lambda_t(x_t) = P(v_t|x_t) \quad (30.9)$$

$$= \underline{x}_t \longrightarrow v_t \quad (30.10)$$

↓
 v_t

30.1 Calculating $P(x_t, v^n)$ and $P(x_t, x_{t+1}, v^n)$

Claim 40 For $t \geq 0$,

$$P(x_t, v^n) = \bar{\mathcal{F}}_t(x_t) \mathcal{F}_t(x_t) \quad (30.11)$$

$$= \sum x_{<t} \longrightarrow x_t \quad x_t \longrightarrow \sum x_{>t} . \quad (30.12)$$

$\downarrow \quad \downarrow \quad \downarrow$
 $v_{<t} \quad v_t \quad v_{>t}$

For $t > 0$,

$$P(x_{t-1}, x_t, v^n) = \bar{\mathcal{F}}_{t-1}(x_{t-1}) P(x_t | x_{t-1}) \lambda_t(x_t) \mathcal{F}_t(x_t) \quad (30.13)$$

$$= \sum x_{<t-1} \longrightarrow x_{t-1} \quad x_{t-1} \longrightarrow x_t \quad x_t \longrightarrow \sum x_{>t} \quad (30.14)$$

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$
 $v_{<t-1} \quad v_{t-1} \quad v_t \quad v_{>t}$

proof:

$$P(x_t, v^n) = \sum_{x_{<i}} \sum_{x_{>i}} P(x^n, v^n) \quad (30.15)$$

$$= \sum_{x_{<i}} \sum_{x_{>i}} P(x^n, v^n | x_t) P(x_t) \quad (30.16)$$

$$= \sum_{x_{<i}} \sum_{x_{>i}} P(x_{<i}, v_{<i}, v_t | x_t) P(x_{>i}, v_{>i} | x_t) P(x_t) \quad (30.17)$$

$$= P(v_{<i}, v_t | x_t) P(v_{>i} | x_t) P(x_t) \quad (30.18)$$

$$= \bar{\mathcal{F}}_t(x_t) \mathcal{F}_t(x_t) \quad (30.19)$$

$$P(x_{t-1}, x_t, v^n) = \sum_{x_{<t-1}} \sum_{x_{>t}} P(x^n, v^n) \quad (30.20)$$

$$= \sum_{x_{<t-1}} \sum_{x_{>t}} P(x^n, v^n | x_{t-1}, x_t) P(x_{t-1}, x_t) \quad (30.21)$$

$$= \sum_{x_{<t-1}} \sum_{x_{>t}} P(x_{<t-1}, v_{<t-1}, v_{t-1} | x_{t-1}) P(v_t | x_t) P(x_{t-1}, x_t) P(x_{>i}, v_{>i} | x_t) \quad (30.22)$$

$$= P(v_{<t-1}, v_{t-1} | x_{t-1}) P(v_t | x_t) P(x_{t-1}, x_t) P(v_{>i} | x_t) \quad (30.23)$$

$$= \bar{\mathcal{F}}_{t-1}(x_{t-1}) \lambda_t(x_t) P(x_t | x_{t-1}) \mathcal{F}_t(x_t) \quad (30.24)$$

QED

30.2 Calculating \mathcal{F}_t and $\overline{\mathcal{F}}_t$

Claim 41 For $t > 0$, \mathcal{F}_t and $\overline{\mathcal{F}}_t$ can be calculated recursively as follows:

$$\overline{\mathcal{F}}_t(x_t) = \sum_{x_{t-1}} \overline{\mathcal{F}}_{t-1}(x_{t-1}) P(x_t|x_{t-1}) \lambda_t(x_t) \quad (30.25)$$

$$= \sum_{x_{t-1}} \sum_{x_{<t-1}} \begin{matrix} x_{<t-1} \longrightarrow x_{t-1} \\ \downarrow \\ v_{<t-1} \end{matrix} \quad \begin{matrix} x_{t-1} \longrightarrow x_t \\ \downarrow \\ v_{t-1} \end{matrix} \quad \begin{matrix} x_t \\ \downarrow \\ v_t \end{matrix} \quad (30.26)$$

and

$$\mathcal{F}_{t-1}(x_{t-1}) = \sum_{x_t} P(x_t|x_{t-1}) \lambda_t(x_t) \mathcal{F}_t(x_t) \quad (30.27)$$

$$= \sum_{x_t} \begin{matrix} x_{t-1} \longrightarrow x_t \\ \downarrow \\ v_t \end{matrix} \quad \begin{matrix} x_t \longrightarrow \sum x_{>t} \\ \downarrow \\ v_{>t} \end{matrix} \quad (30.28)$$

proof:

$$\overline{\mathcal{F}}_t(x_t) \mathcal{F}_t(x_t) = P(x_t, v^n) \quad (30.29)$$

$$= \sum_{x_{t-1}} P(x_{t-1}, x_t, v^n) \quad (30.30)$$

$$= \sum_{x_{t-1}} \overline{\mathcal{F}}_{t-1}(x_{t-1}) \lambda_t(x_t) P(x_t|x_{t-1}) \mathcal{F}_t(x_t) \quad (30.31)$$

$$\overline{\mathcal{F}}_{t-1}(x_{t-1}) \mathcal{F}_{t-1}(x_{t-1}) = P(x_{t-1}, v^n) \quad (30.32)$$

$$= \sum_{x_t} P(x_{t-1}, x_t, v^n) \quad (30.33)$$

$$= \sum_{x_t} \overline{\mathcal{F}}_{t-1}(x_{t-1}) \lambda_t(x_t) P(x_t|x_{t-1}) \mathcal{F}_t(x_t) \quad (30.34)$$

QED

30.3 Calculating $P(x^n|v^n)$

Claim 42

$$P(x_t|x_{t-1}, v^n) = \frac{P(x_t|x_{t-1})\lambda_t(x_t)\mathcal{F}_t(x_t)}{\mathcal{F}_{t-1}(x_{t-1})} \quad (30.35)$$

$$\begin{aligned} & x_{t-1} \longrightarrow x_t \quad x_t \quad x_t \longrightarrow \sum x_{>t} \\ & \downarrow \qquad \qquad \qquad \downarrow \\ & v_t \qquad \qquad \qquad v_{>t} \\ = & \frac{x_{t-1} \longrightarrow \sum x_t \longrightarrow \sum x_{>t}}{x_{t-1} \longrightarrow \sum x_t \longrightarrow \sum x_{>t}} \end{aligned} \quad (30.36)$$

$\downarrow \qquad \qquad \qquad \downarrow$

$v_t \qquad \qquad \qquad v_{>t}$

Note that actually, $P(x_t|x_{t-1}, v^n) = P(x_t|x_{t-1}, v_{\geq t})$ by d-separation, but we won't use this fact.

$$P(x_{t-1}|x_t, v^n) = \frac{\bar{\mathcal{F}}_{t-1}(x_{t-1})P(x_t|x_{t-1})\lambda_t(x_t)}{\bar{\mathcal{F}}_t(x_t)} \quad (30.37)$$

$$\begin{aligned} & \sum x_{<t-1} \longrightarrow x_{t-1} \quad x_{t-1} \longrightarrow x_t \quad x_t \\ & \downarrow \qquad \qquad \qquad \downarrow \qquad \qquad \qquad \downarrow \\ & v_{<t-1} \qquad \qquad v_{t-1} \qquad \qquad v_t \\ = & \frac{\sum x_{<t-1} \longrightarrow \sum x_{t-1} \longrightarrow x_t}{\sum x_{<t-1} \longrightarrow \sum x_{t-1} \longrightarrow x_t} \end{aligned} \quad (30.38)$$

$\downarrow \qquad \qquad \qquad \downarrow \qquad \qquad \qquad \downarrow$

$v_{<t-1} \qquad \qquad v_{t-1} \qquad \qquad v_t$

Note that actually $P(x_{t-1}|x_t, v^n) = P(x_{t-1}|x_t, v_{\leq t-1})$ by d-separation, but we won't use this fact.

proof:

$$P(x_t|x_{t-1}, v^n) = \frac{P(x_{t-1}, x_t, v^n)}{P(x_{t-1}, v^n)} \quad (30.39)$$

$$= \frac{\bar{\mathcal{F}}_{t-1}(x_{t-1})\lambda_t(x_t)P(x_t|x_{t-1})\mathcal{F}_t(x_t)}{\bar{\mathcal{F}}_{t-1}(x_{t-1})\mathcal{F}_{t-1}(x_{t-1})} \quad (30.40)$$

Analogous proof for Eq.(30.38).

QED

$$P(x^n|v^n) = \prod_{t=1,\dots,n-1,n} P(x_t|x_{t-1}, v^n) \quad (\text{forward propagation}) \quad (30.41)$$

$$= \prod_{t=n+1,n,\dots,3,2} P(x_{t-1}|x_t, v^n) \quad (\text{backward propagation}) \quad (30.42)$$

30.4 Calculating $P(v^n|A, B, \pi)$

$P(v^n|A, B, \pi)$ can be calculated

- from $\bar{\mathcal{F}}_{n-1}(x_{n-1})$ (past of x_{n-1})

$$P(v^n|A, B, \pi) = \sum_{x_{n-1}} \underbrace{P(v_{<n-1}, v_{n-1}, x_{n-1})}_{\bar{\mathcal{F}}_{n-1}(x_{n-1})} \quad (30.43)$$

$$= \sum x_{<n-1} \longrightarrow \sum x_{n-1} \quad (30.44)$$

$\downarrow \qquad \qquad \downarrow$

$v_{<n-1} \qquad \qquad v_{n-1}$

- from $\mathcal{F}_0(x_0)$ (future of x_0)

$$P(v^n|A, B, \pi) = \sum_{x_0} P(x_0) P(v_0|x_0) \underbrace{P(v_{>0}|x_0)}_{\mathcal{F}_0(x_0)} \quad (30.45)$$

$$= \sum x_0 \longrightarrow \sum x_{>0} \quad (30.46)$$

$\downarrow \qquad \qquad \downarrow$

$v_0 \qquad \qquad v_{>0}$

- from $\bar{\mathcal{F}}_t(x_t)$ and $\mathcal{F}_t(x_t)$ for some t (past and future of x_t)

$$P(v^n|A, B, \pi) = \sum_{x_t} \underbrace{P(v_{<t}, v_t, x_t)}_{\bar{\mathcal{F}}_t(x_t)} \underbrace{P(v_{>t}|x_t)}_{\mathcal{F}_t(x_t)} \quad (30.47)$$

$$= \sum x_{<t} \longrightarrow \sum x_t \longrightarrow \sum x_{>t} . \quad (30.48)$$

$\downarrow \qquad \qquad \downarrow \qquad \qquad \downarrow$

$v_{<t} \qquad \qquad v_t \qquad \qquad v_{>t}$

30.5 Calculating \hat{x}^n (Viterbi algorithm)

x^n is not visible, only v^n is. Here is how to find an estimate \hat{x}^n of x^n .

Define

$$\bar{\mathcal{F}}_t^{max}(x_t) = \max_{x_{<t}} P(v_{<t}, v_t, x_{<t}, x_t) \quad (30.49)$$

$$= \max x_{<t} \longrightarrow x_t$$

$$\downarrow \qquad \qquad \qquad \downarrow$$

$$v_{<t} \qquad \qquad \qquad v_t$$

$$(30.50)$$

$$\hat{x}_{t-1}(x_t) = \operatorname{argmax}_{x_{t-1}} \bar{\mathcal{F}}_{t-1}^{max}(x_{t-1}) P(x_t|x_{t-1}) \lambda_t(x_t) \quad (30.51)$$

$$= \max x_{<t-1} \longrightarrow \operatorname{argmax}_{x_{t-1}} x_{t-1} \longrightarrow x_t$$

$$\downarrow \qquad \qquad \qquad \downarrow \qquad \qquad \qquad \downarrow$$

$$v_{<t-1} \qquad \qquad \qquad v_{t-1} \qquad \qquad \qquad v_t$$

$$(30.52)$$

For $t > 0$, $\bar{\mathcal{F}}_t^{max}$ and $\hat{x}_{t-1}(x_t)$ can be calculated recursively as follows:

$$\bar{\mathcal{F}}_t^{max}(x_t) = \max_{x_{t-1}} \bar{\mathcal{F}}_{t-1}^{max}(x_{t-1}) P(x_t|x_{t-1}) \lambda_t(x_t) \quad (30.53)$$

$$= \max_{x_{t-1}} \max x_{<t-1} \longrightarrow x_{t-1} \ x_{t-1} \longrightarrow x_t \ x_t$$

$$\downarrow \qquad \qquad \qquad \downarrow \qquad \qquad \qquad \downarrow$$

$$v_{<t-1} \qquad \qquad \qquad v_{t-1} \qquad \qquad \qquad v_t$$

$$(30.54)$$

$$\hat{x}_{t-1}(x_t) = \operatorname{argmax}_{x_{t-1}} \bar{\mathcal{F}}_{t-1}^{max}(x_{t-1}) P(x_t|x_{t-1}) \lambda_t(x_t) \quad (30.55)$$

$$= \operatorname{argmax}_{x_{t-1}} \max x_{<t-1} \longrightarrow x_{t-1} \ x_{t-1} \longrightarrow x_t \ x_t$$

$$\downarrow \qquad \qquad \qquad \downarrow \qquad \qquad \qquad \downarrow$$

$$v_{<t-1} \qquad \qquad \qquad v_{t-1} \qquad \qquad \qquad v_t$$

$$(30.56)$$

Claim 43 (*Viterbi Algorithm*)

If

$$\hat{x}^n = \operatorname{argmax}_{x^n} P(x^n|v^n), \quad (30.57)$$

then the components of \hat{x}^n can be calculated recursively from the last one \hat{x}_{n-1} to the first one \hat{x}_0 , as follows. Let

$$\hat{x}_{n-1} = \operatorname{argmax}_{x_{n-1}} \overline{\mathcal{F}}_{n-1}^{\max}(x_{n-1}) \quad (30.58)$$

$$= \max x_{<n-1} \longrightarrow \operatorname{argmax} x_{n-1} \quad (30.59)$$

\downarrow \downarrow
 $v_{<n-1}$ v_{n-1}

and for $t < n - 1$, use

$$\hat{x}_{t-1} = \hat{x}_{t-1}(\hat{x}_t) \quad (30.60)$$

$$= \max x_{<t-1} \longrightarrow \operatorname{argmax} x_{t-1} \longrightarrow \hat{x}_t \quad (30.61)$$

\downarrow \downarrow \downarrow
 $v_{<t-1}$ v_{t-1} v_t

proof:
QED

30.6 Calculating $\hat{A}, \hat{B}, \hat{\pi}$ (Baum-Welch algorithm)

Let $\theta = (A, B, \pi)$. θ is a set of hidden parameters. Here is how to find an estimate $\hat{\theta}$ of θ .

If x^n and v^n were visible, we could use

$$\hat{\pi}(x) = \mathbb{1}(\underline{x}_0 = x) \quad (30.62)$$

$$\hat{A}(x'|x) = \frac{\sum_{t=0}^{n-2} \mathbb{1}(\underline{x}_t = x, \underline{x}_{t+1} = x')}{\sum_x \text{numerator}} \quad (30.63)$$

$$\hat{B}(v|x) = \frac{\sum_{t=0}^{n-1} \mathbb{1}(\underline{v}_t = v, \underline{x}_t = x)}{\sum_v \text{numerator}} \quad (30.64)$$

But x^n is not visible. So how can we estimate θ under those circumstances?

Define

$$\gamma_t(x_t) = P(x_t|v^n) \quad (30.65)$$

$$= \frac{P(x_t, v^n)}{P(v^n)} \quad (30.66)$$

$$= \frac{\overline{\mathcal{F}}_t(x_t)\mathcal{F}_t(x_t)}{\sum_{x_t} \text{numerator}} \quad (30.67)$$

$$= \frac{\sum x_{<t} \longrightarrow x_t \quad x_t \longrightarrow \sum x_{>t}}{\sum_{x_t} \text{numerator}} \quad (30.68)$$

$$\xi_t(x_t, x_{t+1}) = \frac{P(x_t, x_{t+1}, v^n)}{\sum_{x_t} \sum_{x_{t+1}} \text{numerator}} \quad (30.69)$$

$$= \frac{\overline{\mathcal{F}}_{t-1}(x_{t-1})P(x_t|x_{t-1})\lambda_t(x_t)\mathcal{F}_t(x_t)}{\sum_{x_t} \sum_{x_{t+1}} \text{numerator}} \quad (30.70)$$

$$= \frac{\sum x_{<t-1} \longrightarrow x_{t-1} \quad x_{t-1} \longrightarrow x_t \quad x_t \longrightarrow \sum x_{>t}}{\sum_{x_t} \sum_{x_{t+1}} \text{numerator}} \quad (30.71)$$

Claim 44 (*Baum-Welch algorithm*)

If

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} P(v^n|\theta), \quad (30.72)$$

then we can find $\hat{\theta}$ using the following formulae:

$$\hat{\pi}(x) = \overbrace{\gamma_0(x)}^{\sim P(x)} \quad (30.73)$$

$$\hat{A}(x'|x) = \frac{\sum_{t=0}^{n-2} \overbrace{\xi_t(x, x')}^{\sim P(x, x')}}{\sum_{t=0}^{n-2} \underbrace{\gamma_t(x)}_{P(x)}} \quad (30.74)$$

$$\hat{B}(v|x) = \frac{\sum_{t=0}^{n-1} \overbrace{\mathbb{1}(v_t = v)}^{\sim P(v|x)} \overbrace{\gamma_t(x)}^{\sim P(x)}}{\sum_{t=0}^{n-1} \underbrace{\gamma_t(x)}_{\sim P(x)}} \quad (30.75)$$

proof:

QED

Chapter 31

Influence Diagrams & Utility Nodes

Influence diagrams are just arbitrary bnets enhanced with a new kind of node called an utility node. The rest of this brief chapter will be devoted to discussing utility nodes.

Suppose $U(x)$ is a deterministic function $U : S_x \rightarrow \mathbb{R}$ called the **utility function**. Then the **expected utility** is defined as

$$E_U[U] = \sum_U P(U)U \quad (31.1)$$

$$= \sum_x \sum_U \underbrace{P(U|x)}_{\delta[U,U(x)]} P(x)U \quad (31.2)$$

$$= \sum_x P(x)U(x) . \quad (31.3)$$

An **utility node** can be understood as a node composed of 3 simpler bnet nodes. This is illustrated in Fig.31.1.

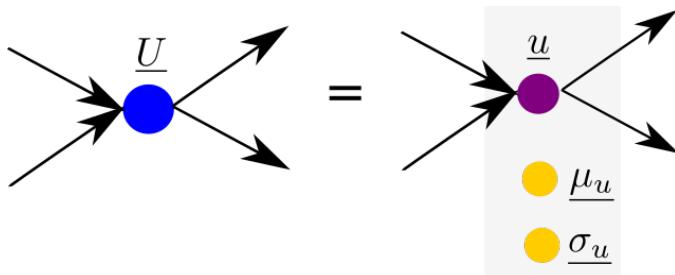


Figure 31.1: An utility node can be understood as a node composed of 3 simpler bnet nodes.

The TPMs, printed in blue, for the bnet Fig.31.1, are as follows:

$$P(U|pa(U)) = \delta[U, U(pa(U))], \quad (31.4)$$

where if $U : S_{\underline{x}} \rightarrow \mathbb{R}$, then $\underline{x} = pa(\underline{U})$.

$$P(u|pa(U)) = \delta[u, U(pa(U))] \quad (31.5)$$

Node $\underline{\mu}_u$ calculates the expected value (mean value) of \underline{u} :

$$P(\mu_u) = \delta(\mu_u, E_{\underline{u}}[\underline{u}]) \quad (31.6)$$

Node $\underline{\sigma}_u$ calculates the standard deviation of \underline{u} :

$$P(\sigma_u) = \delta(\sigma_u, \sqrt{E_{\underline{u}}[(\underline{u} - E_{\underline{u}}[\underline{u}])^2]}) \quad (31.7)$$

Note that in order to calculate expected values, it is necessary that $\underline{U}, \underline{u} \in \mathbb{R}$. Note that nodes $\underline{u}, \underline{\mu}_u, \underline{\sigma}_u$ must all 3 have access to the TPM $P(U|pa(U))$ of node \underline{U} . In fact, in order to calculate $E_{\underline{u}}[\cdot]$, it is necessary for nodes $\underline{\mu}_u$ and $\underline{\sigma}_u$ to have access not just to $P(U|pa(U))$ but also to $P(pa(U))$.

See Fig.31.2. An influence diagram may have multiple utility nodes (\underline{U}_1 and \underline{U}_2 in Fig.31.2). Then one can define a merging utility node \underline{U} that sums the values of all the other utility nodes.

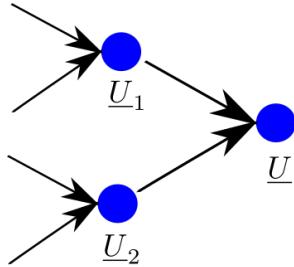


Figure 31.2: An influence diagram may have multiple utility nodes, say \underline{U}_1 and \underline{U}_2 . Then one can define an utility node $\underline{U} = \underline{U}_1 + \underline{U}_2$.

For the node \underline{U} of Fig.31.2,

$$P(U|U_1, U_2) = \delta(U, U_1 + U_2) \quad (31.8)$$

Chapter 32

Instrumental Inequality and beyond

This chapter is based on Refs. [7] and [35].

Instrumental Variables (IVs) are discussed in Chapter 33. This chapter will discuss the original Instrumental inequality (I-inequality) discovered by Pearl, and other related inequalities. The I-inequality arises in bnets that use an IV. The I-inequality bounds the effect that an IV \underline{z} can have on the outcome \underline{y} of a treatment $\underline{d} \rightarrow \underline{y}$. Since there is a path $\underline{z} \rightarrow \underline{d} \rightarrow \underline{y}$, the treatment dose \underline{d} acts as a mediator between the IV \underline{z} and the treatment outcome \underline{y} . The I-inequality is reminiscent of the data processing inequality $H(\underline{z} : \underline{y}) \leq H(\underline{d} : \underline{y})$ which is valid for a simple Markov chain bnet $\underline{z} \rightarrow \underline{d} \rightarrow \underline{y}$. The data processing inequality is saying that the endpoint \underline{y} receives more information from \underline{d} than from \underline{z} . This is reasonable, since \underline{y} is “closer” to \underline{d} than to \underline{z} .

32.1 I-inequality

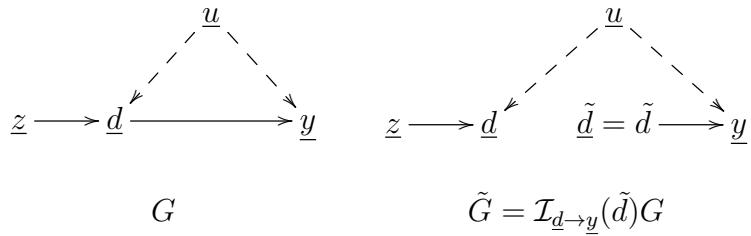


Figure 32.1: In bnet G , an IV \underline{z} acts on a treatment $\underline{d} \rightarrow \underline{y}$. Bnet \tilde{G} is obtained by applying an imagine operator to arrow $\underline{d} \rightarrow \underline{y}$ of bnet G .

Claim 45 *The TPMs for the bnet G in Fig.32.1 satisfy*

$$\max_d \sum_y \max_z P(d, y|z) \leq 1 \quad (32.1)$$

proof:

Below, any probability that alludes to a value \tilde{d} refers to bnet \tilde{G} . Otherwise, if it doesn't allude to \tilde{d} , then it refers to G (or to \tilde{G} , since the TPMs of \tilde{G} are defined from those of G in a consistent manner.)

G satisfies

$$P(d, y|z) = \sum_u P(u)P(y|u, d)P(d|u, z), \quad (32.2)$$

and \tilde{G} satisfies

$$P(d, y|z, \tilde{d}) = \sum_u P(u)P(y|u, \tilde{d})P(d|u, z). \quad (32.3)$$

Note that Eqs.(32.2) and (32.3) imply that

$$P(d, y|z, d) = P(d, y|z) \quad (32.4)$$

and that

$P(\tilde{d}, y|z, \tilde{d}) \leq \sum_d P(d, y|z, \tilde{d}) = P(y|\tilde{d}).$

(32.5)

Thus,

$$\max_{\tilde{d}} \sum_y \max_z P(\tilde{d}, y|z, \tilde{d}) \leq \max_{\tilde{d}} \sum_y \max_z P(y|\tilde{d}) \quad (32.6)$$

$$\leq \max_{\tilde{d}} \sum_y P(y|\tilde{d}) \quad (32.7)$$

$$\leq \max_{\tilde{d}} 1 \quad (32.8)$$

$$\leq 1 \quad (32.9)$$

QED

As pointed out in Ref.[7] from which I learned the above proof, the above proof is highly generalizable.

Fig.32.2 gives a graphical representation of the boxed Eq.(32.5) which is crucial to the proof.

And here is a meta-description of the steps in the proof:

1. Use imagine operator to create a non-negative matrix $M_{d, \tilde{d}}$.
2. Use fact that row or column sum of $M_{d, \tilde{d}}$ is larger than diagonal element in sum:
 $\sum_d M_{d, \tilde{d}} \geq M_{\tilde{d}, \tilde{d}}$.

$$\sum_u \begin{array}{c} \xrightarrow{\quad \underline{u} \quad} \\ \xrightarrow{\quad \tilde{d} = \tilde{d} \quad} \\ \underline{z} \longrightarrow \underline{d} = \tilde{d} \end{array} \leq \sum_d \sum_u \begin{array}{c} \xrightarrow{\quad \underline{u} \quad} \\ \xrightarrow{\quad \tilde{d} = \tilde{d} \quad} \\ \underline{z} \longrightarrow \underline{d} \end{array} = \begin{array}{c} \xrightarrow{\quad \tilde{d} = \tilde{d} \quad} \\ \xrightarrow{\quad \underline{y} \quad} \\ \tilde{d} = \tilde{d} \longrightarrow \underline{y} \end{array} .$$

Figure 32.2: Graphical representation of the boxed equation Eq.(32.5).

32.1.1 I-inequality for binary $\underline{z}, \underline{d}, \underline{y}$

It is enlightening to write down the I-inequality for the special case that $\underline{z}, \underline{d}, \underline{y}$ are binary.

$$P(d=1, y|z) = \begin{matrix} & z=0 & z=1 \\ & \begin{matrix} A & | & B \\ \hline C & | & D \end{matrix} \\ y=0 & & \\ y=1 & & \end{matrix}$$

$$A + D \leq 1$$

$$B + C \leq 1$$

Figure 32.3: I-inequality for binary $\underline{z}, \underline{d}, \underline{y}$. The same picture except with $d = 0$ is also true.

In the binary case, the I-inequality implies 4 different inequalities. These are as follows. One gets two inequalities by setting $d = 1$ in the next 2 equations.

$$\sum_{y=0}^1 \sum_{z=0}^1 \mathbb{1}(y=z) P(d, y|z) , \quad (32.10a)$$

$$\sum_{y=0}^1 \sum_{z=0}^1 \mathbb{1}(y \neq z) P(d, y|z) . \quad (32.10b)$$

One gets an additional 2 inequalities by setting $d = 0$ in Eqs.(32.10). These 4 inequalities are illustrated in Fig.32.3.

What do they mean? That at fixed \underline{d} , the correlation between \underline{z} and \underline{y} is limited.

32.2 Bounds on Effect of IV on treatment outcome \underline{y}

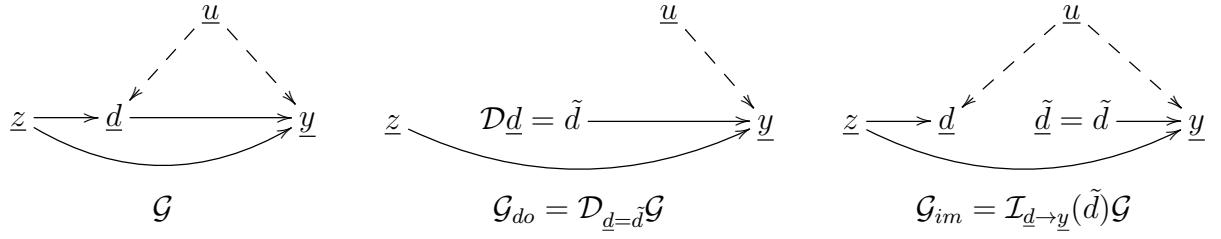


Figure 32.4: Bnet \mathcal{G} is obtained from the bnet G in Fig.32.1 by adding to G an arrow from the IV \underline{z} to the treatment outcome \underline{y} . Bnet \mathcal{G}_{do} is obtained by applying a do operator to node \underline{d} of \mathcal{G} . Bnet \mathcal{G}_{im} is obtained by applying an imagine operator to arrow $\underline{d} \rightarrow \underline{y}$ of \mathcal{G} .

In this section, we will assume that random variables $\underline{z}, \underline{d}, \underline{y}$ are binary. Just as with the binary case of the I-inequality, we will find an inequality for each value of $\underline{d} \in \{0, 1\}$.

Below, we will use the following 3 shorthand notations:

$$P_{y|z}(d) = P(d, y|z) , \quad (32.11)$$

$$P_{|z}(d) = \sum_y P(d, y|z) , \quad (32.12)$$

and

$$\pi_{|z}(d) = 1 - P_{|z}(d) . \quad (32.13)$$

For the bnet \mathcal{G}_{do} in Fig.32.4, define the IV effect at fixed $\mathcal{D}\underline{d} = \tilde{\underline{d}}$ by

$$IVE(\tilde{\underline{d}}) = P(y = 1|z = 1, \mathcal{D}\underline{d} = \tilde{\underline{d}}) - P(y = 1|z = 0, \mathcal{D}\underline{d} = \tilde{\underline{d}}) . \quad (32.14)$$

Claim 46 *The TPMs for the bnet \mathcal{G}_{do} in Fig.32.4 satisfy*

$$\pi_{|0}(d) \leq [IVE(d) - \{P_{1|1}(d) - P_{1|0}(d)\}] \leq \pi_{|1}(d) \quad (32.15)$$

proof:

$$P(y|z, \mathcal{D}\underline{d} = \tilde{d}) = \sum_u P(u)P(y|u, z, \tilde{d}) \quad (32.16)$$

$$= \sum_u P(u) \sum_d P(d, y|u, z, \tilde{d}) \quad (32.17)$$

$$\geq \sum_u P(u)P(\tilde{d}, y|u, z, \tilde{d}) \quad (32.18)$$

$$= \sum_u P(u)P(\tilde{d}, y|u, z) \quad (32.19)$$

$$= P_{y|z}(\tilde{d}) \quad (32.20)$$

Next note that $P(d, y|z, \tilde{d}) \geq 0$, and $\sum_{d,y} P(d, y|z, \tilde{d}) = 1$. If we write a table for $P(d, y|z, \tilde{d})$ at fixed z, \tilde{d} with row and column indices (d, y) , then a partial sum of the entries of that table must be ≤ 1 :

$$\sum_{d \neq \tilde{d}} P(d, y|z, \tilde{d}) + \underbrace{\sum_{y'} P(\tilde{d}, y'|z, \tilde{d})}_{P_{|z}(\tilde{d})} \leq 1 . \quad (32.21)$$

Using the definitions of $P_{|z}$ and $\pi_{|z}$, we can rewrite the last equation as

$$\sum_{d \neq \tilde{d}} P(d, y|z, \tilde{d}) \leq \pi_{|z}(\tilde{d}) . \quad (32.22)$$

Next note that

$$P(y|z, \mathcal{D}\underline{d} = \tilde{d}) = \sum_u P(u)P(y|u, z, \tilde{d}) \quad (32.23)$$

$$= \sum_u P(u) \sum_d P(d, y|u, z, \tilde{d}) \quad (32.24)$$

$$= P(\tilde{d}, y|z, \tilde{d}) + \sum_{d \neq \tilde{d}} P(d, y|z, \tilde{d}) \quad (32.25)$$

$$= P_{y|z}(\tilde{d}) + \sum_{d \neq \tilde{d}} P(d, y|z, \tilde{d}) \quad (32.26)$$

$$\leq P_{y|z}(\tilde{d}) + \pi_{|z}(\tilde{d}) . \quad (32.27)$$

Hence,

$$P_{y|z}(\tilde{d}) \leq P(y|z, \mathcal{D}\underline{d} = \tilde{d}) \leq P_{y|z}(\tilde{d}) + \pi_{|z} \quad (32.28)$$

$$P_{1|1}(\tilde{d}) \leq P(y=1|z=1, \mathcal{D}\underline{d} = \tilde{d}) \leq P_{1|1}(\tilde{d}) + \pi_{|1} \quad (32.29)$$

$$-P_{1|0}(\tilde{d}) - \pi_{|0} \leq -P(y=1|z=0, \mathcal{D}\underline{d} = \tilde{d}) \leq -P_{1|0}(\tilde{d}) \quad (32.30)$$

QED

Chapter 33

Instrumental Variables

This chapter is based on Refs.[6] and [96].

The theory of potential outcomes (PO) discussed in Chapter 56 assumes that confounders can be ignored by conditioning on them. However, there are cases when that is not possible, as when there are some unmeasured (i.e., unobserved, hidden) confounder nodes in the bnet, because one can only condition on observed random variables, by definition. So what if confounders can't be ignored? Are we then precluded from using PO theory? Not necessarily. It might still be possible to use PO theory if one can find a suitable instrumental variable (IV) for the problem.

IVs were actually invented by Sewall Wright and his father Philip Wright long before PO theory was invented by Rubin. The reason why IVs save PO theory is greatly clarified by using Pearl causal DAGs and his d-separation theorem (see Chapter 19).

Most of the discussion in this chapter is limited to LDEN (linear deterministic bnets with external noise). These are discussed in Chapter 38. However, as will become obvious to the reader, IVs are also applicable and useful in general bnet modeling.

In this chapter, for any random variable \underline{x} , we will represent the linear operator $\langle \underline{x}, \cdot \rangle^{-1} \langle \underline{x}, \cdot \rangle$ as a derivative:

$$\frac{d}{d\underline{x}} \cdot = \frac{\langle \underline{x}, \cdot \rangle}{\langle \underline{x}, \underline{x} \rangle} . \quad (33.1)$$

33.1 δ with unmeasured confounder

In this section, we explain using LDENs why unmeasured confounders prejudice PO calculations.

Consider the LDEN bnet of Fig.33.1, For some $\delta, \mu \in \mathbb{R}$, we have

$$\underline{y} = \delta \underline{d} + \underbrace{\mu \underline{h}}_{\underline{n}_y} + \underline{u}_y . \quad (33.2)$$

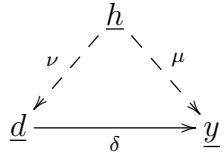


Figure 33.1: An LDEN bnet. The direct path $\underline{d} \rightarrow \underline{y}$ is confounded by a hidden variable \underline{h} .

Note that

$$\langle \underline{y}, \underline{d} \rangle = \delta \langle \underline{d}, \underline{d} \rangle + \langle \underline{n}_{\underline{y}}, \underline{d} \rangle . \quad (33.3)$$

Therefore,

$$\delta = \frac{d\underline{y}}{d\underline{d}} - \frac{d\underline{n}_{\underline{y}}}{d\underline{d}} . \quad (33.4)$$

If the confounder \underline{h} is measured, then we calculate the covariances at fixed $\underline{n}_{\underline{y}}$, and the conditional covariance $\langle \underline{n}_{\underline{y}}, \underline{d} \rangle_{|\underline{n}_{\underline{y}}} = 0$.

33.2 δ (with unmeasured confounder) can be inferred via IV

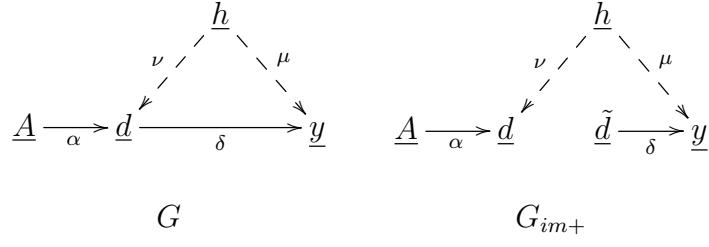


Figure 33.2: Two LDEN bnets. The direct path $\underline{d} \rightarrow \underline{y}$ is confounded by a hidden variable \underline{h} , but by using the IV \underline{A} , we are still able to identify (i.e. calculate) δ .

Now consider the two LDEN bnets shown in Fig.33.2. Note that there are no arrows $\underline{A} \rightarrow \underline{y}$ or $\underline{A} \rightarrow \underline{h}$. Note that node \underline{d} is a collider in the path $\underline{A} - \underline{d} - \underline{h} - \underline{y}$. Therefore, the only unblocked path from \underline{A} to \underline{y} in G is $\underline{A} \rightarrow \underline{d} \rightarrow \underline{y}$ and that path has been removed in G_{im+} . These observations are encapsulated in the following statements.

$$\underline{d} \perp_G \underline{y} = \text{false}, \quad \underline{A} \perp_G \underline{y} = \text{false} . \quad (33.5)$$

$$\underline{d} \perp_{G_{im+}} \underline{y} = \text{false}, \quad \underline{A} \perp_{G_{im+}} \underline{y} = \text{true} . \quad (33.6)$$

The following is true for G :

$$\underline{y} = \delta \underline{d} + \underbrace{\mu \underline{h} + \underline{u}_y}_{\underline{n}_y} \quad (33.7)$$

$$\underline{d} = \alpha \underline{A} + \underbrace{\nu \underline{h} + \underline{u}_d}_{\underline{n}_d} . \quad (33.8)$$

Since $\langle \underline{n}_y, \underline{A} \rangle = \langle \underline{n}_d, \underline{A} \rangle = 0$ is true in G , we have

$$\langle \underline{y}, \underline{A} \rangle = \delta \langle \underline{d}, \underline{A} \rangle \quad (33.9)$$

and

$$\langle \underline{d}, \underline{A} \rangle = \alpha \langle \underline{A}, \underline{A} \rangle . \quad (33.10)$$

Note that $\langle \underline{y}, \underline{A} \rangle = \delta = 0$ for G_{im+} but not for G , so we are speaking about G from here on. It follows that

$$\alpha = \frac{d\underline{d}}{d\underline{A}} \quad (33.11)$$

and

$$\delta = \frac{\langle \underline{y}, \underline{A} \rangle}{\langle \underline{d}, \underline{A} \rangle} \quad (33.12)$$

$$= \frac{\langle \underline{y}, \underline{A} \rangle}{\langle \underline{A}, \underline{A} \rangle} \frac{\langle \underline{A}, \underline{A} \rangle}{\langle \underline{d}, \underline{A} \rangle} \quad (33.13)$$

$$= \frac{\frac{dy}{d\underline{A}}}{\frac{d\underline{d}}{d\underline{A}}} \quad (\text{equal to } \frac{dy}{d\underline{d}}) \quad (33.14)$$

$$= \frac{dy}{d(\alpha \underline{A})} . \quad (33.15)$$

Eq.(33.14) is illustrated in Fig.33.3.

33.3 More general bnets with IVs

Figs.33.4 and 33.5 are examples of other bnets for which the effect δ is identifiable thanks to the IV \underline{A} .

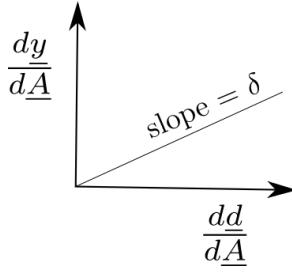


Figure 33.3: Effect $\delta = \frac{dy}{d\underline{d}}$ as slope of line.

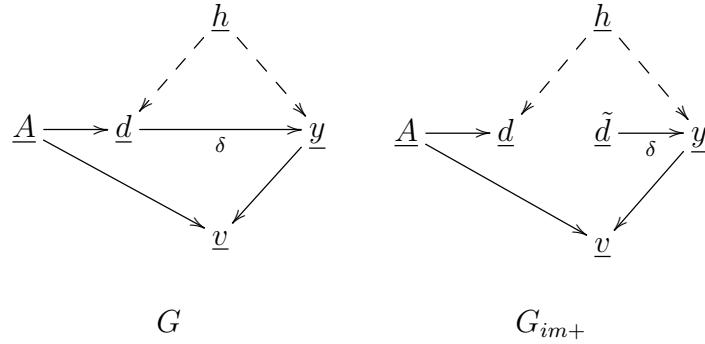


Figure 33.4: The 2 paths in G_{im+} from IV variable \underline{A} to \underline{y} are blocked by not conditioning on colliders \underline{v} and \underline{d} . Thus, $\underline{d} \perp_{G_{im+}} \underline{y} = \text{false}$, $\underline{A} \perp_{G_{im+}} \underline{y} = \text{true}$

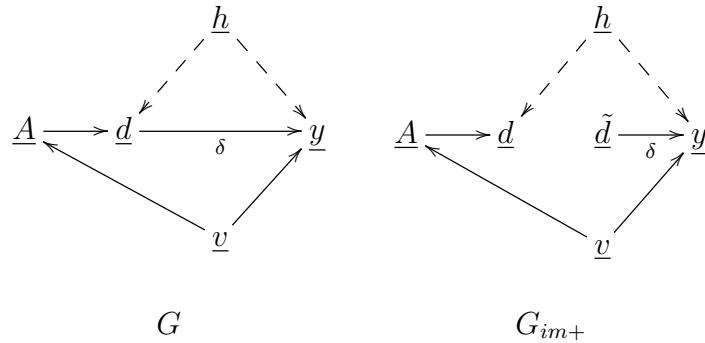


Figure 33.5: There are 2 paths in G_{im+} from IV variable \underline{A} to \underline{y} . One is blocked by not conditioning on the collider \underline{d} and the other can be blocked by conditioning on \underline{v} . Thus, $\underline{d} \perp_{G_{im+}} \underline{y} | \underline{v} = \text{false}$, $\underline{A} \perp_{G_{im+}} \underline{y} | \underline{v} = \text{true}$

33.4 Instrumental Inequality

Pearl's instrumental inequality and related inequalities are discussed in Chapter 32.

Chapter 34

Jackknife Resampling

This paper is based on Ref. [98].

Before reading this chapter, we recommend that you read the section entitled “Demystifying Population and Sample Variances”, in Chapter Notational Conventions and Preliminaries.

Jackknife Resampling (JR) is a way of generating from an original list of n samples, a new list of n synthetic (i.e., man made, not occurring physically in Nature) samples obtained by deleting one of the samples from the original list and averaging over the rest.

Let $\Sigma = \{0, 1, 2, \dots, n - 1\}$. Let us consider the list of samples

$$\vec{x} = (x^\sigma)_{\sigma \in \Sigma}, \quad (34.1)$$

where the x^σ are assumed to be i.i.d. with

$$E[\underline{x}^\sigma] = \mu \quad (34.2)$$

and

$$\left\langle \underline{x}^\sigma, \underline{x}^{\sigma'} \right\rangle = V_1 \delta(\sigma, \sigma'). \quad (34.3)$$

If we define μ and V_1 estimators by

$$\hat{\mu} = \frac{1}{n} \sum_{\sigma} x^\sigma \quad (34.4a)$$

$$\hat{V}_1 = \frac{1}{n-1} \sum_{\sigma} (x^\sigma - \hat{\mu})^2, \quad (34.4b)$$

then one can show that:

$$E[\hat{\mu}] = \mu, \quad E[\hat{V}_1] = V_1 \quad (34.5)$$

so both of these estimators are unbiased.

Now define lists of samples with one of the items in \vec{x} deleted:

$$\vec{x}_\xi = (x^\sigma)_{\sigma \in \Sigma - \{\xi\}} . \quad (34.6)$$

Suppose we are given functions of \vec{x} and \vec{x}_ξ

$$A = A^n(\vec{x}) , \quad (34.7)$$

$$A_\xi = A^{n-1}(\vec{x}_\xi) . \quad (34.8)$$

Then define a list \vec{A} by

$$\vec{A} = (A_\xi)_{\xi \in \Sigma} . \quad (34.9)$$

One can also define a list \vec{B} by:

$$\vec{B} = (B_\xi)_{\xi \in \Sigma} \quad (34.10)$$

where

$$B_\xi = nA - (n-1)A_\xi \quad (34.11)$$

$$= A_\xi - n[A_\xi - A] . \quad (34.12)$$

Later on, we will see why the list \vec{B} just defined is of interest.

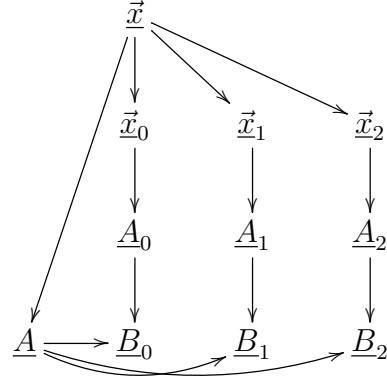


Figure 34.1: Bnet for jackknife resampling (JR).

Fig.34.1 is a bnet that encapsulates JR. The TPMs, printed in blue, for that bnet, are as follows:

$$P(\vec{x}_\xi | \vec{x}) = \mathbb{1}(\vec{x}_\xi = \text{defined by Eq.(34.6)}) \quad (34.13)$$

$$P(A_\xi | \vec{x}_\xi) = \mathbb{1}(\quad A_\xi = \text{ defined by Eq.(34.8)} \quad) \quad (34.14)$$

$$P(B_\xi | \vec{A}_\xi, A) = \mathbb{1}(\quad B_\xi = \text{ defined by Eq.(34.12)} \quad) \quad (34.15)$$

34.1 Case $A = A^n(\vec{x}) = \frac{1}{n} \sum_{\sigma} x^{\sigma}$

Suppose

$$A = \frac{1}{n} \underbrace{\sum_{\sigma} x^{\sigma}}_{E_{\sigma}[x^{\sigma}]} \quad (34.16)$$

and

$$A_\xi = \underbrace{\frac{1}{n-1} \sum_{\sigma \in \Sigma - \{\xi\}} x^{\sigma}}_{E_{\sigma|\xi}[x^{\sigma}] \text{ where } P(\sigma|\xi) = \frac{\mathbb{1}(\sigma \neq \xi)}{n-1}} \quad . \quad (34.17)$$

Then

$$\frac{1}{n} \sum_{\xi} A_\xi = E_{\xi} E_{\sigma|\xi}[x^{\sigma}] = E_{\sigma}[x^{\sigma}] = A \quad (34.18)$$

Claim 47

$$E[A_\xi] = \mu \quad (34.19)$$

$$\langle \underline{A}_\xi, \underline{A}_{\xi'} \rangle = V_1 \left[\frac{n-2-\delta(\xi, \xi')}{n-1} \right] \quad (34.20)$$

proof:

$$E[\underline{A}_\xi] = \frac{1}{n-1} \sum_{\sigma} \mathbb{1}(\xi \neq \sigma) E[x^{\sigma}] = \mu \quad (34.21)$$

$$\langle \underline{A}_\xi, \underline{A}_{\xi'} \rangle = \frac{1}{n-1} \sum_{\sigma} \sum_{\sigma'} [1 - \delta(\sigma, \xi)] [1 - \delta(\sigma', \xi')] \langle x^{\sigma}, x^{\sigma'} \rangle \quad (34.22)$$

$$= \frac{V_1}{n-1} \sum_{\sigma} \sum_{\sigma'} [1 - \delta_{\xi}^{\sigma} - \delta_{\xi'}^{\sigma'} + \delta_{\xi}^{\sigma} \delta_{\xi'}^{\sigma'}] \delta_{\sigma'}^{\sigma} \quad (34.23)$$

$$= \frac{V_1}{n-1} [n-2 + \delta_{\xi'}^{\xi}] \quad (34.24)$$

QED

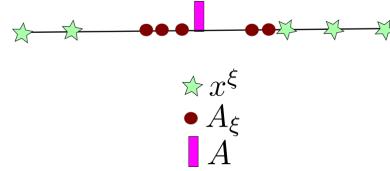


Figure 34.2: For each point x^ξ , there is a corresponding point A_ξ which is $n - 1$ times closer to the average A than x^ξ is.

Claim 48

$$A_\xi - A = \frac{A - x^\xi}{n - 1} \quad (34.25)$$

Hence, the distance of a point x^ξ to the mean value A is $n - 1$ times as large as the distance of A_ξ to A . (see Fig.34.2)

proof:

$$A_\xi - A = \frac{1}{n - 1} \left(\sum_{\sigma} x^\sigma - x^\xi \right) - \frac{1}{n} \sum_{\sigma} x^\sigma \quad (34.26)$$

$$= x^\xi \left(\frac{-1}{n - 1} \right) + \sum_{\sigma} x^\sigma \underbrace{\left(\frac{1}{n - 1} - \frac{1}{n} \right)}_{\frac{1}{n(n-1)}} \quad (34.27)$$

$$= \frac{A - x^\xi}{n - 1} \quad (34.28)$$

QED

Note that

$$(n - 1) \sum_{\xi} (A_\xi - E_{\xi}[A_\xi])^2 = \hat{V}_1 \quad (34.29)$$

by Eq.(34.25). Hence, we can estimate V_1 from \vec{A} instead of \vec{x} .

Note that

$$B_\xi = nA - (n - 1)A_\xi \quad (34.30)$$

$$= \sum_{\sigma} x^\sigma - \sum_{\sigma \neq \xi} x^\sigma = x^\xi \quad (34.31)$$

Since $B_\xi = x^\xi$, they have identical statistics. In particular, one can use for B_ξ the same μ and V_1 estimators that we defined in Eqs.(34.4) for x^σ .

Chapter 35

Junction Tree Algorithm

The Junction Tree (JT) algorithm is an algo for evaluating exact marginals of a bnet, including cases in which some nodes are fixed to a single state. (fixed nodes are called the a priori evidence.)

The JT algo starts by clustering the loops of a bnet into bigger nodes so as to transform the bnet into a polytree bnet. Then it applies Pearl Belief Propagation (see Chapter 43) to the ensuing polytree. The first breakthrough paper to achieve this agenda in full was Ref.[20] by Lauritzen, and Spiegelhalter in 1988. See the Wikipedia article Ref.[99] for more info and references on the JT algorithm.

I won't describe the JT algo any further here, because it would take too long for this brief book to give a complete treatment of it, including the mathematical proofs. If all you want to do is to code the JT algo, without delving into the mathematical theorems and proofs behind it, I strongly recommend Ref.[16]. Ref.[16] is an excellent cookbook for programmers of the JT algo. My open source program QuantumFog (see Ref.[62]) implements the JT algo in Python, following the recipe of Ref.[16].

Chapter 36

Kalman Filter

This chapter is based on Ref.[101], except we've replaced the variables F_t, w_t, v_t in that reference by A_t, ξ_t, ζ_t , respectively.

A Kalman Filter (KF) is a special case of a Hidden Markov Model. HMMs are discussed in Chapter 30.

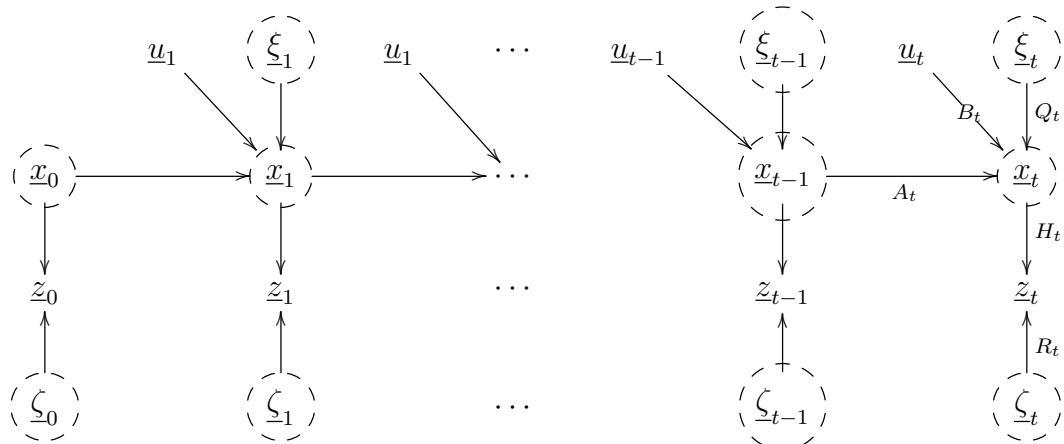


Figure 36.1: Kalman Filter (KF) bnet.

Let

$t = 0, 1, 2, \dots, T - 1$ be the time.

$\underline{\xi}_t \in \mathbb{R}^{nx}$, $\underline{\zeta}_t \in \mathbb{R}^{nz}$ be random variables that represent hidden (unobserved) external Gaussian white noise.

$\underline{x}_t \in \mathbb{R}^{nx}$ be random variables that represent the hidden (unobserved) true state of the system.

$\underline{u}_t \in \mathbb{R}^{nu}$, $\underline{z}_t \in \mathbb{R}^{nz}$ be random variables that represent the measured (observed) state of the system.

The TPMs, printed in blue, for the KF bnet Fig.36.1, are as follows:

$$P(\xi_t) = \mathcal{N}(\xi_t; 0, Q_t), \quad (36.1)$$

where Q_t is given.

$$P(x_t|x_{t-1}, u_t, \xi_t) = \mathbb{1}(x_t = A_t x_{t-1} + B_t u_t + \xi_t), \quad (36.2)$$

where $A_t, B_t u_t$ are given. $P(x_t|x_{t-1}, u_t, \xi_t)$ becomes $P(x_t)$ for $t = 0$.

$$P(\zeta_t) = \mathcal{N}(\zeta_t; 0, R_t), \quad (36.3)$$

where R_t is given.

$$P(z_t|x_t, \zeta_t) = \mathbb{1}(z_t = H_t x_t + \zeta_t), \quad (36.4)$$

where H_t is given.

36.1 Prediction Problem

Find \hat{x}_t (the best possible estimate of x_t) and P_t (the state of the filter at time t) in terms of

1. \hat{x}_{t-1} and P_{t-1} .
2. the 5 matrices $\mathcal{M}_t = (A_t, B_t, H_t, Q_t, R_t)$
3. the observed values of z_t and u_t .

See Fig.36.2. For that figure,

$$P(\hat{x}_t, P_t | \hat{x}_{t-1}, P_{t-1}, \mathcal{M}_t, z_t, u_t) = \delta(\hat{x}_t, ?) \delta(P_t, ?). \quad (36.5)$$

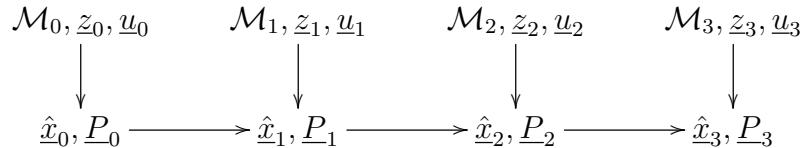


Figure 36.2: Evolution of \hat{x}_t, P_t for a KF.

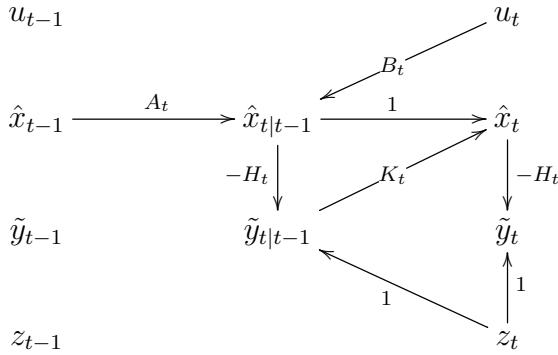
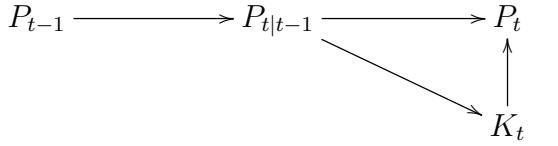


Figure 36.3: Bnet representation of the algebraic solution of the prediction problem for a KF.

36.2 Solution

The algebraic solution of the prediction problem for a KF is as follows. See Fig.36.3 for a bnet representation of this algebraic solution.

Define $\eta_{t|t} = \eta_t$ for $\eta = \hat{x}, P$.

- **Initial Conditions** \hat{x}_0, P_0

- **A priori estimates**

a priori state estimate

$$\hat{x}_{t|t-1} = \underbrace{A_t \hat{x}_{t-1} + B_t u_t}_{x_t - \xi_t} \quad (36.6)$$

a priori covariance estimate

$$P_{t|t-1} = A_t P_{t-1} A_t^T + Q_t \quad (36.7)$$

- **A posteriori estimates**

Optimal Kalman gain K_t

$$S_t = H_t P_{t|t-1} H_t^T + R_t \quad (36.8)$$

$$K_t = P_{t|t-1} H_t^T S_t^{-1} \quad (36.9)$$

$$= P_{t|t-1} H_t^T [H_t P_{t|t-1} H_t^T + R_t]^{-1} \quad (36.10)$$

a posteriori state estimate

$$\tilde{y}_{t|t-1} = z_t - H_t \hat{x}_{t|t-1} \quad (36.11)$$

$$\hat{x}_t = \hat{x}_{t|t-1} + K_t \tilde{y}_{t|t-1} \quad (36.12)$$

$$= (1 - K_t H_t) \hat{x}_{t|t-1} + K_t \underbrace{z_t}_{H_t x_t + \zeta_t} \quad (\text{interpolation formula}) \quad (36.13)$$

$$\tilde{y}_t = z_t - H_t \hat{x}_t \quad (36.14)$$

a posteriori covariance estimate

$$P_t = (I - K_t H_t) P_{t|t-1} \quad (36.15)$$

36.3 Simple Example

r_t position, v_t velocity, a_t acceleration of point particle.

$$x_t = Ax_{t-1} + Bu_t + \xi_t \quad (36.16)$$

$$x_t = \begin{bmatrix} r_t \\ v_t \end{bmatrix}, \quad u_t = a_t \quad (36.17)$$

$$A = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} \frac{1}{2}(\Delta t)^2 \\ \Delta t \end{bmatrix} \quad (36.18)$$

$$z_t = Hx_t + \zeta_t \quad (36.19)$$

$$H = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad (36.20)$$

36.4 Invariants

Note that

$$x_t - \hat{x}_{t|t-1} = \xi_t \quad (36.21)$$

$$x_t - \hat{x}_t = (1 - K_t H_t)(x_t - \hat{x}_{t|t-1}) - K_t \zeta_t \quad (36.22)$$

$$= (1 - K_t H_t)\xi_t - K_t \zeta_t \quad (36.23)$$

$$\tilde{y}_{t|t-1} = \underbrace{z_t}_{H_t x_t + \zeta_t} - H_t \hat{x}_{t|t-1} \quad (36.24)$$

$$= H_t \xi_t + \zeta_t \quad (36.25)$$

$$\tilde{y}_t = \underbrace{z_t}_{H_t x_t + \zeta_t} - H_t \hat{x}_t \quad (36.26)$$

$$= H_t(x_t - \hat{x}_t) + \zeta_t \quad (36.27)$$

$$= H_t(1 - K_t H_t)\xi_t + (1 - H_t K_t)\zeta_t \quad (36.28)$$

$(x_t - \hat{x}_{t|t-1})$, $(x_t - \hat{x}_t)$, $\tilde{y}_{t|t-1}$ and \tilde{y}_t are called **residuals**. Since ξ_t and ζ_t have zero mean value,

$$E[x_t - \hat{x}_t] = E[x_t - \hat{x}_{t|t-1}] = 0 \quad (36.29)$$

$$E[\tilde{y}_t] = E[\tilde{y}_{t|t-1}] = 0 \quad (36.30)$$

These zero mean value identities are called **invariants**.

36.5 Derivation of Solution

First, some notational conventions. Let

$$Cov(\underline{a})_{i,j} = \langle \underline{a}_i, \underline{a}_j \rangle \quad (36.31)$$

$$Cov(\underline{a}) = \langle \underline{a}, \underline{a}^T \rangle = \langle \underline{a}, tp. \rangle \quad (36.32)$$

$$[A, B]_+ = AB + B^T A^T \quad (36.33)$$

$$A + tp. = A + A^T \quad (36.34)$$

tp. stands for transpose.

Now define

$$P_t = Cov(\underline{x}_t - \hat{\underline{x}}_t) \quad (36.35)$$

$$P_{t|t-1} = Cov(\underline{x}_t - \hat{\underline{x}}_{t|t-1}) \quad (36.36)$$

$$S_t = Cov(\tilde{y}_{t|t-1}) \quad (36.37)$$

It follows that

$$P_t = \langle \underline{x}_t - \hat{\underline{x}}_t, tp. \rangle \quad (36.38)$$

$$= \left\langle (1 - K_t H_t)(\underline{x}_t - \hat{\underline{x}}_{t|t-1}) - K_t \underline{\zeta}_t, tp. \right\rangle \quad (\text{by Eq.(36.23)}) \quad (36.39)$$

$$= \left\langle (1 - K_t H_t)(\underline{x}_t - \hat{\underline{x}}_{t|t-1}), tp. \right\rangle + \left\langle K_t \underline{\zeta}_t, tp. \right\rangle \quad (\underline{\zeta}_t \text{ uncorrelated with } (\underline{x}_t - \hat{\underline{x}}_{t|t-1})) \quad (36.40)$$

$$= (1 - K_t H_t) \underbrace{\langle \underline{x}_t - \hat{\underline{x}}_{t|t-1}, tp. \rangle}_{P_{t|t-1}} (1 - K_t H_t)^T + K_t \underbrace{\langle \underline{\zeta}_t, tp. \rangle}_{R_t} K_t^T \quad (36.41)$$

$$= P_{t|t-1} - [K_t H_t, P_{t|t-1}]_+ + K_t \underbrace{(H_t P_{t|t-1} H_t^T + R_t)}_{S_t} K_t^T \quad (36.42)$$

Next we find the optimal Kalman gain K_t by minimizing with respect to K_t the following mean squared error.

$$\mathcal{E} = \sum_i E[(\underline{x}_t - \hat{\underline{x}}_t)_i^2] \quad (36.43)$$

$$= \text{tr} E[(\underline{x}_t - \hat{\underline{x}}_t)(\underline{x}_t - \hat{\underline{x}}_t)^T] \quad (36.44)$$

$$= \text{tr} P_t \quad (36.45)$$

$$= \text{tr}(P_{t|t-1} - [K_t H_t, P_{t|t-1}]_+ + K_t S_t K_t^T) \quad (36.46)$$

If we set to zero the variation of \mathcal{E} when K_t varies, we get

$$0 = \delta \mathcal{E} = \text{tr} [(-P_{t|t-1} H_t^T + K_t S_t) \delta K_t^T] + tp. \quad (36.47)$$

Hence

$$-P_{t|t-1} H_t^T + K_t S_t = 0 \quad (36.48)$$

$$K_t = P_{t|t-1} H_t^T S_t^{-1} \quad (36.49)$$

Chapter 37

Linear and Logistic Regression

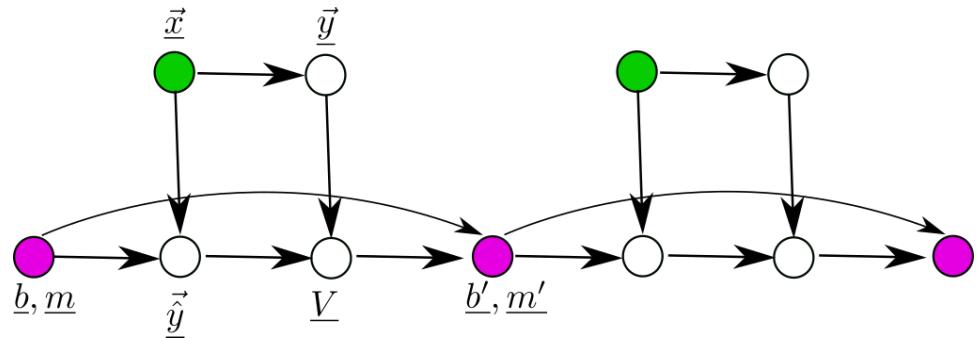


Figure 37.1: Linear Regression

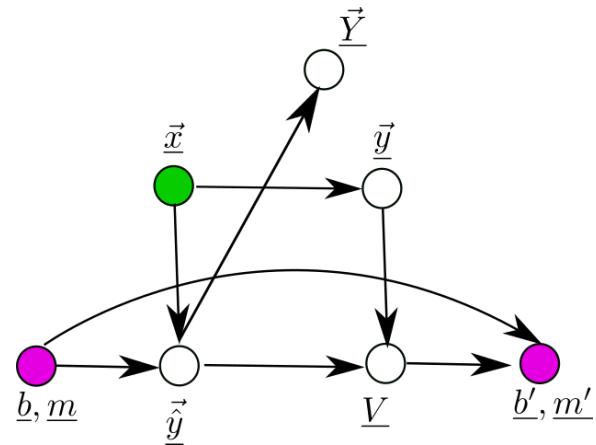


Figure 37.2: Bnet of Fig.37.1 with new \vec{Y} node.

Estimators \hat{y} for linear and logistic regression.

- **Linear Regression:** $y \in \mathbb{R}$. Note $\hat{y} \in \mathbb{R}$. $(x, \hat{y}(x))$ is the graph of a straight

line with y-intercept b and slope m .

$$\hat{y}(x; b, m) = b + mx \quad (37.1)$$

- **Logistic Regression:** $y \in \{0, 1\}$. Note $\hat{y} \in [0, 1]$. $(x, \hat{y}(x))$ is the graph of a sigmoid. Often in literature, b, m are replaced by β_0, β_1 .

$$\hat{y}(x; b, m) = \text{smoid}(b + mx) \quad (37.2)$$

Define

$$V(b, m) = \sum_{x,y} P(x, y) |y - \hat{y}(x; b, m)|^2. \quad (37.3)$$

We want to minimize $V(b, m)$ (called a cost or loss function) wrt b and m .

The TPMs, printed in blue, for the Bnet Fig.37.1, are as follows.

$$P(b, m) = \text{given} \quad (37.4)$$

The first time it is used, (b, m) is arbitrary. After the first time, it is determined by previous stage.

Let

$$P_{\underline{x}, \underline{y}}(x, y) = \frac{1}{nsam(\vec{x})} \sum_{\sigma} \mathbb{1}(x = x^{\sigma}, y = y^{\sigma}). \quad (37.5)$$

$$P(\vec{x}) = \prod_{\sigma} P(x^{\sigma}) \quad (37.6)$$

$$P(\vec{y}|\vec{x}) = \prod_{\sigma} P(y^{\sigma} | x^{\sigma}) \quad (37.7)$$

$$P(\vec{\hat{y}}|\vec{x}, b, m) = \prod_{\sigma} \delta(\hat{y}^{\sigma}, \hat{y}(x^{\sigma}, b, m)) \quad (37.8)$$

$$P(V|\vec{\hat{y}}, \vec{y}) = \delta(V, \frac{1}{nsam(\vec{x})} \sum_{\sigma} |y^{\sigma} - \hat{y}^{\sigma}|^2) \quad (37.9)$$

Let $\eta_b, \eta_m > 0$. For $x = b, m$, if $x' - x = \Delta x = -\eta \frac{\partial V}{\partial x}$, then $\Delta V \approx \frac{-1}{\eta} (\Delta x)^2 \leq 0$ for $\eta > 0$. This is called “gradient descent”.

$$P(b'|V, b) = \delta(b', b - \eta_b \partial_b V) \quad (37.10)$$

$$P(m'|V, m) = \delta(m', m - \eta_m \partial_m V) \quad (37.11)$$

37.1 Generalization to x with multiple components (features)

Suppose that for each sample σ , instead of x^σ being a scalar, it has n components called features:

$$x^\sigma = (x_0^\sigma, x_1^\sigma, x_2^\sigma, \dots, x_{n-1}^\sigma) . \quad (37.12)$$

Slope m is replaced by weights

$$w = (w_0, w_1, w_2, \dots, w_{n-1}) , \quad (37.13)$$

and the product of 2 scalars mx^σ is replaced by the inner vector product $w^T x^\sigma$.

37.2 Alternative $V(b, m)$ for logistic regression

For logistic regression, since $y^\sigma \in \{0, 1\}$ and $\hat{y}^\sigma \in [0, 1]$ are both in the interval $[0, 1]$, they can be interpreted as probabilities. Define probability distributions $p^\sigma(x)$ and $\hat{p}^\sigma(x)$ for $x \in \{0, 1\}$ by

$$p^\sigma(1) = y^\sigma, \quad p^\sigma(0) = 1 - y^\sigma \quad (37.14)$$

$$\hat{p}^\sigma(1) = \hat{y}^\sigma, \quad \hat{p}^\sigma(0) = 1 - \hat{y}^\sigma \quad (37.15)$$

Then for logistic regression, the following 2 cost functions $V(b, m)$ can be used as alternatives to the cost function Eq.(37.3) previously given.

$$V(b, m) = \frac{1}{nsam(\vec{x})} \sum_{\sigma} D_{KL}(p^\sigma \| \hat{p}^\sigma) \quad (37.16)$$

and

$$V(b, m) = \frac{1}{nsam(\vec{x})} \sum_{\sigma} CE(p^\sigma \| \hat{p}^\sigma) \quad (37.17)$$

$$= \frac{-1}{nsam(\vec{x})} \sum_{\sigma} \{y^\sigma \ln \hat{y}^\sigma + (1 - y^\sigma) \ln(1 - \hat{y}^\sigma)\} \quad (37.18)$$

$$= \frac{-1}{nsam(\vec{x})} \sum_{\sigma} \ln \{(\hat{y}^\sigma)^{y^\sigma} (1 - \hat{y}^\sigma)^{(1-y^\sigma)}\} \quad (37.19)$$

$$= \frac{-1}{nsam(\vec{x})} \sum_{\sigma} \ln P(\underline{Y} = y^\sigma | \hat{y} = \hat{y}^\sigma) \quad (37.20)$$

$$= - \sum_{x,y} P(x, y) \ln P(\underline{Y} = y | \hat{y} = \hat{y}(x, b, m)) \quad (37.21)$$

Above, we used

$$P(\underline{Y} = Y | \hat{y}) = \hat{y}^Y [1 - \hat{y}]^{1-Y} \quad (37.22)$$

for $Y \in S_{\underline{Y}} = \{0, 1\}$. (Bernoulli distribution).

There is no node corresponding to \underline{Y} in the Bnet of Fig.37.1. Fig.37.2 shows a new Bnet that has a new node called $\vec{\underline{Y}}$ compared to the Bnet of Fig.37.1. One defines the TPMs for all nodes of Fig.37.2 except $\vec{\underline{Y}}$ and \underline{V} the same as for Fig.37.1. For $\vec{\underline{Y}}$ and \underline{V} , one defines

$$P(Y^\sigma | \vec{\hat{y}}) = P(\underline{Y} = Y^\sigma | \hat{y}^\sigma) \quad (37.23)$$

$$P(V|\vec{\underline{Y}}, \vec{y}) = \delta(V, \frac{-1}{nsam(\vec{x})} \ln \mathcal{L}), \quad (37.24)$$

where $\mathcal{L} = \prod_\sigma P(\underline{Y} = y^\sigma | \hat{y}^\sigma)$ = likelihood.

Chapter 38

Linear Deterministic Bnets with External Noise

In this chapter, we will consider bnets which were referred to, prior to the invention of bnets, as: Sewall Wright's **Path Analysis (PA)** and **linear Structural Equations Models (SEM)**. Judea Pearl in his books calls them **linear Structural Causal Models (SCM)**, because they are very convenient for doing causal analysis. We will refer to them as linear Deterministic with External Noise (LDEN) diagrams. This chapter is devoted to LDEN diagrams, except that we will say a few words about non-linear DEN diagrams at the end.

A **DEN diagram** is a special kind of bnet. To build a DEN diagram, start with a deterministic bnet G . The deterministic nodes of G are called the **endogenous (internal) variables**. Now make a bigger bnet \bar{G} called a DEN diagram by adding to each node a of G a non-deterministic root node u_a pointing into a only. The nodes u_a are called the **exogenous (external) variables**. The exogenous variables make their children noisy. They are assumed to be unobserved and their TPMs are prior probability distributions. Since they are root nodes, they are mutually independent. When we draw a DEN diagram, we will never draw the exogenous nodes, leaving them implicit.

A **linear DEN diagram (LDEN)** is a DEN diagram whose deterministic nodes have a TPM that is a linear function of the states of the parent nodes.

38.1 Example of LDEN diagram

The TPMs, printed in blue, for the LDEN diagram Fig.38.1, are as follows.

$$P(y|w, z, u_y) = \mathbb{1}(y = \epsilon w + \delta z + u_y) \quad (38.1)$$

$$P(w|x, z, u_w) = \mathbb{1}(w = \beta x + \gamma z + u_w) \quad (38.2)$$

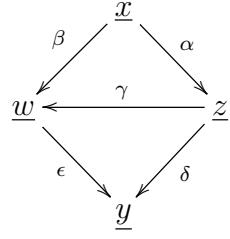


Figure 38.1: Example of a LDEN diagram wherein \underline{x} splits into two nodes \underline{z} and \underline{w} , then merges into node \underline{y} . There is also an arrow $\underline{z} \rightarrow \underline{w}$. Exogenous nodes are not shown. The Greek letters represent real numbers.

$$P(z|x, u_{\underline{z}}) = \mathbb{1}(z = \alpha x + u_{\underline{z}}) \quad (38.3)$$

$$P(x|u_{\underline{x}}) = \mathbb{1}(x = u_{\underline{x}}) \quad (38.4)$$

Hence,

$$y = \epsilon w + \delta z + u_{\underline{y}} \quad (38.5)$$

$$= \epsilon(\beta x + \gamma z + u_{\underline{w}}) + \delta z + u_{\underline{y}} \quad (38.6)$$

$$= (\epsilon\gamma + \delta)z + \epsilon\beta x + \epsilon u_{\underline{w}} + u_{\underline{y}} \quad (38.7)$$

$$= (\epsilon\gamma + \delta)z + \epsilon\beta u_{\underline{x}} + \epsilon u_{\underline{w}} + u_{\underline{y}} . \quad (38.8)$$

Therefore

$$\left(\frac{\partial y}{\partial z} \right)_{u_{\cdot} - u_{\underline{z}}} = \epsilon\gamma + \delta , \quad (38.9)$$

where the partial derivative holds fixed all exogenous variables except $u_{\underline{z}}$. Note that this partial derivative is a sum of terms, and that each of those terms represents a different directed path from \underline{z} to $\underline{y}(z)$. This is a general property of LDEN diagrams.

38.2 Fully Connected LDEN diagrams

The bnets that will be considered in this section will all be fully connected. Fully connected bnets are defined in Chapter Definition of a Bayesian Network. This section uses the notation $\langle \underline{x}, \underline{y} \rangle$ for the covariance of any two random variables $\underline{x}, \underline{y}$. This $\langle \underline{x}, \underline{y} \rangle$ notation is defined in Chapter Notational Conventions and Preliminaries.

Consider a LDEN diagram with deterministic nodes $\underline{x} = (\underline{x}_k)_{k=0,1,\dots,nx-1}$ and corresponding exogenous nodes $\underline{u} = (\underline{u}_k)_{k=0,1,\dots,nx-1}$. Assume $\langle \underline{u}_i, \underline{u}_j \rangle = 0$ if $i \neq j$. The strength of each connection $\underline{x}_i \rightarrow \underline{x}_j$ of the LDEN diagram is measured by a **structural coefficient** $\alpha_{j|i} \in \mathbb{R}$. Some of the $\alpha_{j|i}$ may be zero, in which case the corresponding arrow $i \rightarrow j$ would not be drawn.

38.2.1 Fully connected LDEN diagram with $nx = 2$

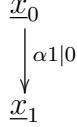


Figure 38.2: Fully connected LDEN diagram with two \underline{x}_j nodes (exogenous nodes \underline{u}_j not shown).

Consider the LDEN diagram of Fig.38.2. This diagram represents the following **structural equations**:

$$\underline{x}_0 = \underline{u}_0 \quad (38.10a)$$

$$\underline{x}_1 = \alpha_{1|0}\underline{x}_0 + \underline{u}_1. \quad (38.10b)$$

Eqs.38.10 constitute a system of 2 linear equations in 2 unknowns (the \underline{x} 's) so we can solve for the \underline{x} 's in terms of the α 's and \underline{u} 's.

Note also that

$$\langle \underline{x}_1, \underline{x}_0 \rangle = \alpha_{1|0} \langle \underline{x}_0, \underline{x}_0 \rangle. \quad (38.11)$$

Thus, $\alpha_{1|0}$ can be estimated from the covariances $\langle \underline{x}_i, \underline{x}_j \rangle$.

38.2.2 Fully connected LDEN diagram with $nx = 3$

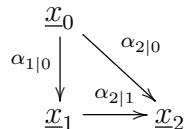


Figure 38.3: Fully connected LDEN diagram with three \underline{x}_j nodes (exogenous nodes \underline{u}_j not shown).

Consider the LDEN diagram of Fig.38.3. This diagram represents the following **structural equations**:

$$\underline{x}_0 = \underline{u}_0 \quad (38.12a)$$

$$\underline{x}_1 = \alpha_{1|0}\underline{x}_0 + \underline{u}_1 \quad (38.12b)$$

$$\underline{x}_2 = \alpha_{2|0}\underline{x}_0 + \alpha_{2|1}\underline{x}_1 + \underline{u}_2 . \quad (38.12c)$$

Eqs.38.12 constitute a system of 3 linear equations in 3 unknowns (the \underline{x} 's) so we can solve for the \underline{x} 's in terms of the α 's and \underline{u} 's.

Note also that

$$\langle \underline{x}_1, \underline{x}_0 \rangle = \alpha_{1|0} \langle \underline{x}_0, \underline{x}_0 \rangle \quad (38.13a)$$

$$\langle \underline{x}_2, \underline{x}_0 \rangle = \alpha_{2|0} \langle \underline{x}_0, \underline{x}_0 \rangle + \alpha_{2|1} \langle \underline{x}_1, \underline{x}_0 \rangle \quad (38.13b)$$

$$\langle \underline{x}_2, \underline{x}_1 \rangle = \alpha_{2|0} \langle \underline{x}_0, \underline{x}_1 \rangle + \alpha_{2|1} \langle \underline{x}_1, \underline{x}_1 \rangle \quad (38.13c)$$

Eqs.38.13 constitute a system of 3 linear equations in 3 unknowns (the α 's) so we can solve for the α 's in terms of covariances $\langle \underline{x}_i, \underline{x}_j \rangle$. This gives an estimate for the α 's.

38.2.3 Fully connected LDEN diagram with arbitrary nx

Let $\underline{x}_. = (x_i)_{i=0,1,\dots,nx-1}$ and $\underline{x}_{<i} = (x_k)_{k=0,1,\dots,i-1}$. Consider a fully connected LDEN diagram with deterministic nodes labeled \underline{x}_i . The \underline{x}_i labels are assumed to be in **topological order** (i.e., the parents of node \underline{x}_i are $\underline{x}_{<i}$). Let the TPMs, printed in blue, for the nodes $\underline{x}_.$ of the LDEN diagram, be as follows.

$$P(x_i|\underline{x}_{<i}, \underline{u}_i) = \mathbb{1}(x_i = \sum_{k< i} \alpha_{i|k} x_k + u_i) , \quad (38.14)$$

for some parameters $\alpha_{i|k} \in \mathbb{R}$. The exogenous nodes $\underline{u}_.$ are assumed to be independent so

$$P(\underline{u}_.) = \prod_i P(u_i) \quad (38.15)$$

and

$$\langle \underline{u}_i, \underline{u}_j \rangle = 0 \text{ if } i \neq j . \quad (38.16)$$

Note that

$$P(x_.) = \sum_{\underline{u}_.} P(\underline{u}_.) \prod_i P(x_i|\underline{x}_{<i}, u_i) \quad (38.17)$$

$$= E_{\underline{u}_.} [\prod_i P(x_i|\underline{x}_{<i}, u_i)] . \quad (38.18)$$

In terms of random variables, this system is described by the following **structural equations**:

$$\underline{x}_i = \sum_{k < i} \alpha_{i|k} \underline{x}_k + \underline{u}_i . \quad (38.19)$$

The structural equations can be written in matrix form as follows. Define a strictly lower triangular matrix A with the connection strengths $\alpha_{i|k} \in \mathbb{R}$ as entries. For example, for $nx = 4$,

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \alpha_{1|0} & 0 & 0 & 0 \\ \alpha_{2|0} & \alpha_{2|1} & 0 & 0 \\ \alpha_{3|0} & \alpha_{3|1} & \alpha_{3|2} & 0 \end{bmatrix} . \quad (38.20)$$

If we now represent the multinodes $\underline{x}_.$ and $\underline{u}_.$ as column vectors \underline{x} and \underline{u} , we get

$$\underline{x} = A \underline{x} + \underline{u} . \quad (38.21)$$

Note that

$$\underline{x} = (1 - A)^{-1} \underline{u} . \quad (38.22)$$

Therefore,

$$\underline{x}_i = f_i(\underline{u}_{\leq i}) . \quad (38.23)$$

Therefore, if $i > j$,

$$\langle \underline{u}_i, \underline{x}_j \rangle = \langle \underline{u}_i, f_j(\underline{u}_{\leq j}) \rangle = 0 . \quad (38.24)$$

Thus, if $i > j$,

$$\langle \underline{x}_i, \underline{x}_j \rangle = \sum_{k < i} \alpha_{i|k} \langle \underline{x}_k, \underline{x}_j \rangle + \langle \underline{u}_i, \underline{x}_j \rangle \quad (38.25)$$

$$= \sum_{k < i} \alpha_{i|k} \langle \underline{x}_k, \underline{x}_j \rangle . \quad (38.26)$$

In matrix notation, Eq.(38.26) becomes

$$\langle \underline{x}, \underline{x}^T \rangle_L = A[\langle \underline{x}, \underline{x}^T \rangle_L + \langle \underline{x}, \underline{x}^T \rangle_D] \quad (38.27)$$

where we are using $\langle \underline{x}, \underline{x}^T \rangle_{i,j} = \langle \underline{x}_i, \underline{x}_j \rangle$ and denoting the strictly lower triangular part and diagonal part of a matrix M by M_L and M_D . Thus,

$$A = \langle \underline{x}, \underline{x}^T \rangle_L [\langle \underline{x}, \underline{x}^T \rangle_L + \langle \underline{x}, \underline{x}^T \rangle_D]^{-1} . \quad (38.28)$$

This gives an estimate for the α 's in terms of the covariances $\langle \underline{x}_i, \underline{x}_j \rangle$.

38.3 Non-linear DEN diagrams

This chapter is dedicated to linear DEN diagrams. This implicitly assumes that the deterministic nodes $\underline{x}_.$ of the DEN diagram have an interval of real values as their possible states. A trivial but very useful generalization of linear DEN diagrams is to replace Eq.(38.14) for the TPMs of the deterministic nodes of the diagram by

$$P(x_i|x_{<i}, u_i) = \mathbb{1}(x_i = f_i(x_{<i}, u_i)) , \quad (38.29)$$

with structural equations

$$\underline{x}_i = f_i(\underline{x}_{<i}, \underline{u}_i) , \quad (38.30)$$

for $i = 0, 1, \dots, nx - 1$. Here the f_i are possibly non-linear functions that depend the states $x_{<i}$ and u_i of nodes $\underline{x}_{<i}$ and \underline{u}_i . If a node \underline{x}_i has no arrows entering it (i.e., is a root node), then

$$P(x_i|x_{<i}, u_i) = P(x_i) = \delta(x_i, a) \quad (38.31)$$

and

$$\underline{x}_i = a \quad (38.32)$$

for some $a \in S_{\underline{x}_i}$.

With this generalization, we can make any $f_i()$ represent a continuous probability distribution such as a Gaussian, or a discrete-valued Boolean function such as an OR gate.

Eqs.(38.29) and (38.30) are the TPMs and structural equations for a fully connected, non-linear DEN diagram. For a non-fully connected diagram,

- replace the multinode $x_{<i}$ by a subset of itself, in Eqs.(38.29) and (38.30) , and
- delete the corresponding arrows from the graph.

Chapter 39

Markov Blankets

This chapter is based on the Wikipedia article, Ref.[108]. Markov blankets and Markov boundaries of bnets were apparently invented by Judea Pearl. His 1988 book Ref.[37], instead of a research paper, is usually given as the original reference.

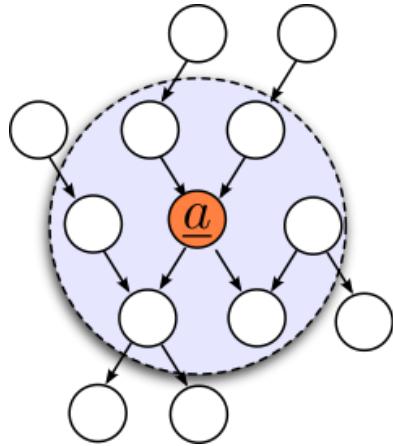


Figure 39.1: In a bnet, the minimal Markov blanket, aka Markov boundary, of node \underline{a} .

We will treat vectors of random variables as if they were sets when using the \in , \subset and $-$ operations. For example, if $\underline{x} = (\underline{x}_0, \underline{x}_1, \underline{x}_2, \underline{x}_3)$ and $\underline{b} = (\underline{x}_1, \underline{x}_2)$, then $\underline{x}_1 \in \underline{b} \subset \underline{x}$ and $\underline{x} - \underline{b} = (\underline{x}_0, \underline{x}_3)$.

Below, $H(\underline{a} : \underline{b} | \underline{c})$ denotes the conditional mutual information of random variables \underline{a} and \underline{b} conditioned on random variable \underline{c} . $H(\underline{a} : \underline{b} | \underline{c})$ is used in Shannon Information Theory, where it is defined by

$$H(\underline{a} : \underline{b} | \underline{c}) = \sum_{a,b,c} P(a, b, c) \ln \frac{P(a, b | c)}{P(a | c)P(b | c)}. \quad (39.1)$$

$H(\underline{a} : \underline{b} | \underline{c}) = 0$ iff \underline{a} and \underline{b} are independent (uncorrelated) when \underline{c} is held fixed.

Suppose $\underline{a} \in \underline{X}$, $\underline{B} \subset \underline{X}$, but $\underline{a} \notin \underline{B}$. Then \underline{B} is a **Markov blanket** of \underline{a} if

$$H(\underline{a} : \underline{X} - \underline{a} | \underline{B}) = 0 . \quad (39.2)$$

In other words, one may assume that \underline{a} depends on \underline{B} only, and is independent of all random variables in $\underline{X} - (\underline{a} \cup \underline{B})$.

The **minimal Markov blanket** is called the **Markov boundary**.

In a bnet, the Markov boundary of a node \underline{a} , contains:

1. the parents of \underline{a} ,
2. the children of \underline{a} ,
3. the parents, other than \underline{a} , of the children of \underline{a} .

This is illustrated in Fig.39.1.

From the d-separation theorem (see Chapter 19), we get an intuitive motivation for the definition of Markov boundary. By conditioning on the parents and children of node \underline{a} , we block almost all, but not all, information from reaching node \underline{a} . The reason not all info is blocked is that conditioning on a child \underline{c} of \underline{a} creates paths from a parent of \underline{c} to \underline{c} to \underline{a} , wherein node \underline{c} acts as a conditioned collider. To block such paths, we must condition on the parents of \underline{c} too.

Chapter 40

Markov Chain Monte Carlo (MCMC)

Monte Carlo methods are methods for using random number generation to sample probability distributions. The subject of Monte Carlo methods has many branches, as you can see from its Wikipedia category list, Ref.[111]. MCMC (Markov Chain Monte Carlo) is just one of those branches, albeit a major one. Metropolis-Hastings (MH) sampling is a very important MCMC method. Gibbs sampling is a special case of MH sampling. This chapter covers both, MH and Gibbs sampling. It also covers a few other types of sampling.

Throughout this chapter, we use $P_{\underline{x}} : S_{\underline{x}} \rightarrow [0, 1]$ to denote the target probability distribution that we wish to obtain samples from.

40.1 Inverse Cumulative Sampling

For more info about this topic and some original references, see Ref.[97].

This is one of the simplest methods for obtaining samples from a probability distribution $P_{\underline{x}}$, but it requires knowledge of the inverse cumulative distribution of $P_{\underline{x}}$, which is often not available.

The **cumulative distribution** function is defined by:

$$CUM_{\underline{x}}(x) = P(\underline{x} < x) = \int_{x' < x} dx' P_{\underline{x}}(x') . \quad (40.1)$$

Note that

$$P_{\underline{x}}(x) = \frac{d}{dx} CUM_{\underline{x}}(x) . \quad (40.2)$$

For $t = 0, 1, \dots, T - 1$, let

$\underline{u}^{(t)} \in [0, 1]$ = random variable, uniformly distributed over $[0, 1]$.

$\vec{\underline{x}}^{(t)} = (\underline{x}^{(t)}[\sigma])_{\sigma=0,1,\dots,nsam(t)-1}$ where $\underline{x}^{(t)}[\sigma] \in S_{\underline{x}}$ for all σ . Vector of samples collected up to time t .

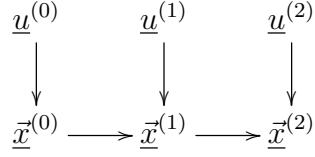


Figure 40.1: bnet for Inverse Cumulative Sampling

The TPMs, printed in blue, for bnet Fig.40.1, are as follows:

$$P(\underline{u}^{(t)}) = 1 \quad (40.3)$$

$$P(\underline{x}^{(t)} | \underline{x}^{(t-1)}, \underline{u}^{(t)}) = \delta(\underline{x}^{(t)}, [\underline{x}^{(t-1)}, CUM_{\underline{x}}^{-1}(\underline{u}^{(t)})]) \quad (40.4)$$

Motivation

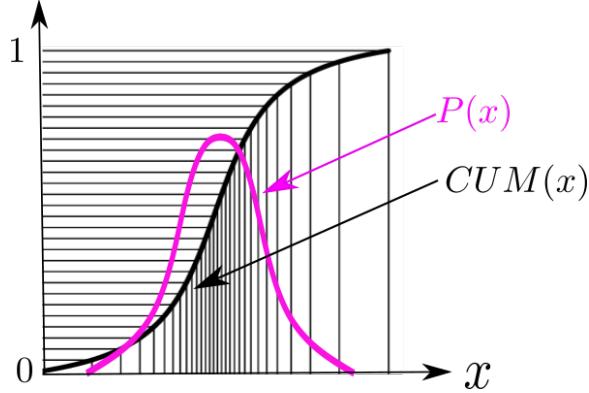


Figure 40.2: Motivation for Inverse Cumulative Sampling.

See Fig.40.2.

Note that if \underline{u} is uniformly distributed over the interval $[0, 1]$ and $a \in [0, 1]$, then

$$P(\underline{u} < a) = a . \quad (40.5)$$

Thus

$$P(CUM_{\underline{x}}^{-1}(\underline{u}) < x) = P(\underline{u} < CUM_{\underline{x}}(x)) \quad (40.6)$$

$$= CUM_{\underline{x}}(x) . \quad (40.7)$$

Therefore,

$$dP(CUM_{\underline{x}}^{-1}(\underline{u}) < x) = P_{\underline{x}}(x)dx . \quad (40.8)$$

40.2 Rejection Sampling

For more info about this topic and some original references, see Ref.[122].

This method samples from a “candidates” probability distribution $P_c : S_x \rightarrow [0, 1]$, in cases where sampling directly from the target probability distribution $P_{\underline{x}} : S_{\underline{x}} \rightarrow [0, 1]$ is not possible.

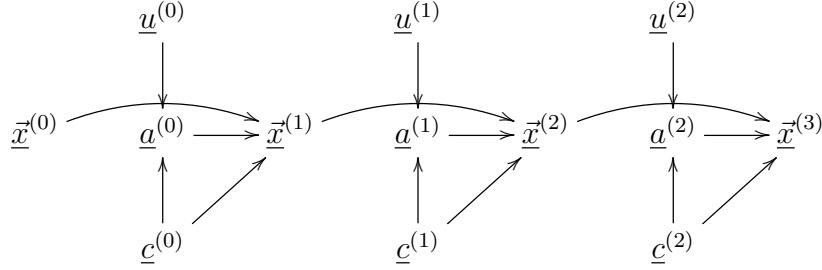


Figure 40.3: bnet for Rejection Sampling

For $t = 0, 1, \dots, T - 1$, let

$\underline{u}^{(t)} \in [0, 1]$ = random variable, uniformly distributed over $[0, 1]$.

$\underline{a}^{(t)} \in \{0, 1\}$ = accept candidate? (no=0, yes=1)

$\underline{c}^{(t)} \in S_x$ = sample that is a candidate for being accepted

$\vec{x}^{(t)} = (\underline{x}^{(t)}[\sigma])_{\sigma=0,1,\dots,nsam(t)-1}$ where $\underline{x}^{(t)}[\sigma] \in S_x$ for all σ . Vector of samples collected up to time t .

This algorithm requires a priori definition of a candidate probability distribution $P_c : S_x \rightarrow \mathbb{R}$ such that

$$P_{\underline{x}}(x) < \beta P_c(x) \quad (40.9)$$

for all $x \in S_x$, for some $\beta \in \mathbb{R}$.

The TPMs, printed in blue, for bnet Fig.40.3, are as follows:

$$P(u^{(t)} = u) = 1 \quad (40.10)$$

$$P(\underline{c}^{(t)} = c) = P_c(c) \quad (40.11)$$

$$P(\underline{a}^{(t)} = a | \underline{c}^{(t)} = c, \underline{u}^{(t)} = u) = \begin{cases} \delta(a, 0) & \text{if } u\beta P_c(c) \geq P_{\underline{x}}(c) \\ \delta(a, 1) & \text{if } u\beta P_c(c) < P_{\underline{x}}(c) \end{cases} \quad (40.12)$$

$$P(\vec{x}^{(t)} | \vec{x}^{(t-1)}, \underline{a}^{(t)} = a, \underline{c}^{(t)} = c) = \begin{cases} \delta(\vec{x}^{(t)}, \vec{x}^{(t-1)}) & \text{if } a = 0 \\ \delta(\vec{x}^{(t)}, [\vec{x}^{(t-1)}, c]) & \text{if } a = 1 \end{cases} \quad (40.13)$$

This last equation is only defined for $t > 0$. For $t = 0$, the left hand side reduces to $P(\vec{x}^{(0)})$ which must be specified a priori.

Motivation

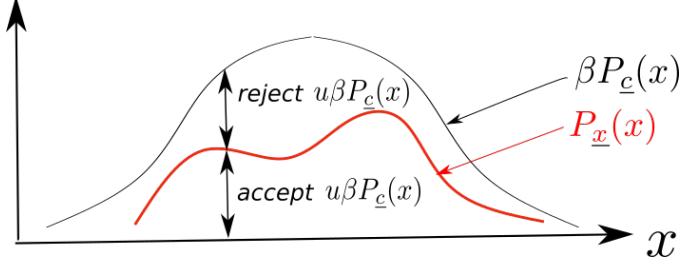


Figure 40.4: Motivation for Rejection Sampling.

See Fig.40.4.

40.3 Metropolis-Hastings Sampling

For more info about this topic and some original references, see Refs.[2] and [109].

An advantage of this method is that it can sample unnormalized probability distributions (*constant*) $P_{\underline{x}}$ because it only uses ratios of $P_{\underline{x}}$ at two different points. Another advantage of this method is that it scales much better than other sampling methods as the number of dimensions of the sampled variable \underline{x} increases.

This method produces samples that take a finite amount of time to reach steady state. The samples are also theoretically correlated instead of being i.i.d. as one desires. To mitigate for the steady state problem, one discards an initial set of samples (the “burn-in” period). To mitigate for the correlation problem, one calculates the autocorrelation between the samples and keeps only samples separated by a time interval after which the samples cease to be autocorrelated to a good approximation.

For $t = 0, 1, \dots, T - 1$, let

$\underline{u}^{(t)} \in [0, 1]$ = random variable, uniformly distributed over $[0, 1]$.

$\underline{a}^{(t)} \in \{0, 1\}$ = accept candidate? (no=0, yes=1)

$\underline{c}^{(t)} \in S_{\underline{x}}$ = sample that is a candidate for being accepted

$\underline{m}^{(t)} \in S_{\underline{x}}$ = memory of last accepted sample

$\vec{\underline{x}}^{(t)} = (\underline{x}^{(t)}[\sigma])_{\sigma=0,1,\dots,nsam(t)-1}$ where $\underline{x}^{(t)}[\sigma] \in S_{\underline{x}}$ for all σ . Vector of samples collected up to time t .

A **proposal TPM** $P_{c|\underline{x}} : S_{\underline{x}}^2 \rightarrow [0, 1]$ must be specified a priori for this algorithm.

The TPMs, printed in blue, for bnet Fig.40.5, are as follows:

$$P(u^{(t)} = u) = 1 \quad (40.14)$$

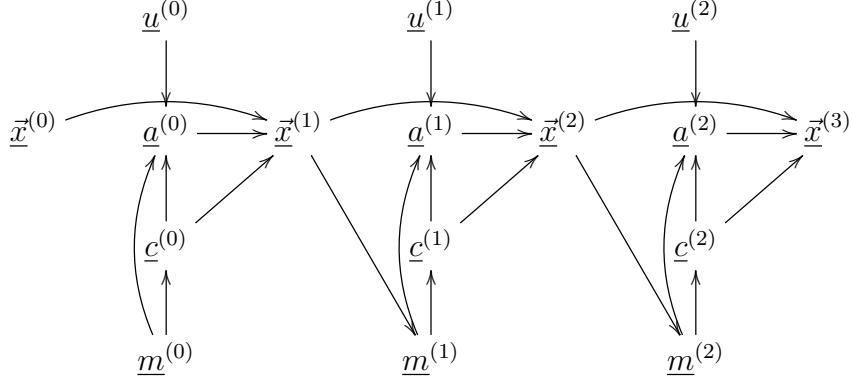


Figure 40.5: bnet for Metropolis-Hastings Sampling

$$P(\underline{c}^{(t)} = c | \underline{m}^{(t)} = m) = P_{\underline{c}|\underline{x}}(c|m) \quad (40.15)$$

$$P(\underline{a}^{(t)} = a | \underline{c}^{(t)} = c, \underline{u}^{(t)} = u, \underline{m}^{(t)} = m) = \begin{cases} \delta(a, 0) & \text{if } u \geq \alpha(c|m) \\ \delta(a, 1) & \text{if } u < \alpha(c|m) \end{cases} \quad (40.16)$$

where the **acceptance probability** α is defined as

$$\alpha(c|m) = \min \left(1, \frac{P_{\underline{c}|\underline{x}}(m|c) P_{\underline{x}}(c)}{P_{\underline{c}|\underline{x}}(c|m) P_{\underline{x}}(m)} \right). \quad (40.17)$$

Note that if the proposal distribution is symmetric, then

$$\alpha(c|m) = \min \left(1, \frac{P_{\underline{x}}(c)}{P_{\underline{x}}(m)} \right). \quad (40.18)$$

$$P(\vec{x}^{(t)} | \vec{x}^{(t-1)}, \underline{a}^{(t)} = a, \underline{c}^{(t)} = c) = \begin{cases} \delta(\vec{x}^{(t)}, \vec{x}^{(t-1)}) & \text{if } a = 0 \\ \delta(\vec{x}^{(t)}, [\vec{x}^{(t-1)}, c]) & \text{if } a = 1 \end{cases} \quad (40.19)$$

This last equation is only defined for $t > 0$. For $t = 0$, the left hand side reduces to $P(\vec{x}^{(0)})$ which must be specified a priori.

$$P(\underline{m}^{(t)} = m | \vec{x}^{(t)}) = \delta(m, \text{last component of } \vec{x}^{(t)}). \quad (40.20)$$

This last equation is only defined for $t > 0$. For $t = 0$, the left hand side reduces to $P(\underline{m}^{(0)} = m)$ which must be specified a priori.

Motivation

See Fig.40.6.

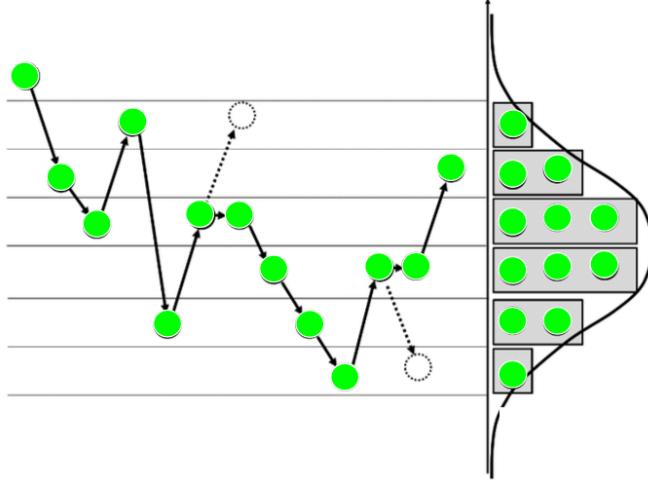


Figure 40.6: Motivation for Metropolis-Hastings Sampling.

Consider a time homogeneous (its TPM is the same for all times) Markov chain with TPM $P(x'|x) = [T]_{x',x}$. Its **stationary distribution**, if it exists, is defined as

$$\pi = \lim_{n \rightarrow \infty} T^n \pi_0 . \quad (40.21)$$

Suppose the prob distribution $P_{\underline{x}}(x)$ that we wish to sample from satisfies

$$P_{\underline{x}}(x) = \pi(x) . \quad (40.22)$$

Reversibility (detailed balance): For all $x, x' \in S_{\underline{x}}$,

$$P(x'|x)\pi(x) = P(x|x')\pi(x') . \quad (40.23)$$

Detailed balance is a sufficient (although not necessary) condition for a unique stationary prob distribution π to exist.¹

Let

$$P(x'|x) = P(\underline{a} = 1|x', x)P_{c|x}(x'|x) + \delta(x, x')P(\underline{a} = 0|x) , \quad (40.24)$$

where

$$P(\underline{a} = 0|x) = \sum_{x'} P(\underline{a} = 0|x', x)P_{c|x}(x'|x) . \quad (40.25)$$

Claim 49 *If*

¹As explained lucidly in Ref.[2], besides detailed balance, 2 other properties must also be satisfied by the Markov chain, irreducibility and aperiodicity. However, because of how it is constructed, the Metropolis-Hastings algorithm automatically produces a Markov chain that has those 2 properties.

$$P(\underline{a} = 1|x', x) = \alpha(x'|x), \quad (40.26)$$

then detailed balance is satisfied.

proof: Assume $x \neq x'$.

$$P(x'|x)P(x) = P(\underline{a} = 1|x', x)P_{\underline{c}|\underline{x}}(x'|x)P_{\underline{x}}(x) \quad (40.27)$$

$$= \min \left(1, \frac{P_{\underline{c}|\underline{x}}(x|x')P_{\underline{x}}(x')}{P_{\underline{c}|\underline{x}}(x'|x)P_{\underline{x}}(x)} \right) P_{\underline{c}|\underline{x}}(x'|x)P_{\underline{x}}(x) \quad (40.28)$$

$$= \min (P_{\underline{c}|\underline{x}}(x'|x)P_{\underline{x}}(x), P_{\underline{c}|\underline{x}}(x|x')P_{\underline{x}}(x')) \quad (40.29)$$

$$= P(x|x')P(x') \quad (40.30)$$

QED

40.4 Gibbs Sampling

For more info about this topic and some original references, see Ref.[91].

Gibbs sampling is a special case of Metropolis-Hastings sampling. Gibbs sampling is ideally suited for application to a bnet, because it is stated in terms of the conditional prob distributions of N random variables, and conditional prob distributions are part of the definition of a bnet.

Consider a bnet with nodes $\underline{x}_0, \underline{x}_1, \dots, \underline{x}_{N-1}$

Identify the random variable $\underline{x} = (\underline{x}_0, \underline{x}_1, \dots, \underline{x}_{N-1})$ with the random variable \underline{x} used in Metropolis-Hastings sampling. For Gibbs sampling, we use the following proposal distribution:

$$P_{\underline{c}|\underline{x}}(c|m) = \prod_{j=0}^{N-1} P(c_j | [m_i]_{i \neq j}). \quad (40.31)$$

Eq.(40.31) can be simplified using Markov Blankets (see Chapter 39) to the following:

$$P_{\underline{c}|\underline{x}}(c|m) = \prod_{j=0}^{N-1} P(c_j | [m_i : \forall i \ni \underline{x}_i \in MB(\underline{x}_j)]), \quad (40.32)$$

where, for any node \underline{a} , we denote its Markov blanket by $MB(\underline{a})$.

An alternative proposal distribution that leads to much faster convergence is as follows. The idea is to make the components $c_j^{(t)}$ of candidate sample $c^{(t)}$ depend on the previous components $(c_i^{(t)})_{i < j}$. See the bnet Fig.40.7. The TPM for the nodes of that bnet are

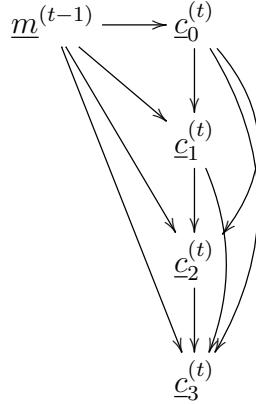


Figure 40.7: In Gibbs sampling, the proposal distribution $P_{\underline{c}|\underline{x}}$ can be defined by making the components $c_j^{(t)}$ of candidate sample $c^{(t)}$ depend on the previous components $(c_i^{(t)})_{i < j}$.

$$P(\underline{c}_j^{(t)} = c_j | (\underline{c}_i^{(t)})_{i < j} = (c_i)_{i < j}, \underline{m}^{(t-1)} = m) = P(c_j | (c_i)_{i < j}, (m_i)_{i > j}) \quad (40.33)$$

for $j = 0, 1, \dots, N - 1$. This implies

$$P_{\underline{c}|\underline{x}}(\underline{c}^{(t)} = c | \underline{m}^{(t-1)} = m) = \prod_{j=0}^{N-1} P(c_j | (c_i)_{i < j}, (m_i)_{i > j}) . \quad (40.34)$$

As before, we can condition only on the Markov blanket of each node \underline{x}_j .

$$P_{\underline{c}|\underline{x}}(\underline{c}^{(t)} = c | \underline{m}^{(t-1)} = m) = \prod_{j=0}^{N-1} P(c_j | (c_i)_{i < j}, (m_i)_{i > j}, \text{ use only } c_i \text{ and } m_i \ni \underline{x}_i \in MB(\underline{x}_j)) . \quad (40.35)$$

40.5 Importance Sampling

For more info about this topic and some original references, see Ref.[95].

Suppose random variables $\underline{x}[\sigma] \in S_{\underline{x}}$ for $\sigma = 0, 1, \dots, nsam - 1$ are i.i.d. with probability distribution $P_{\underline{x}}$. Then

$$E_{\underline{x}}[f(x)] \approx \frac{1}{nsam} \sum_{\sigma=0}^{nsam-1} f(x[\sigma]) \quad (40.36)$$

for any $f : S_{\underline{x}} \rightarrow \mathbb{R}$. Sometimes, instead of using i.i.d. samples $\underline{x}[\sigma] \in S_{\underline{x}}$ where $\underline{x}[\sigma] \sim P_{\underline{x}}$, we wish to use i.i.d. samples $\underline{y}[\sigma] \in S_{\underline{x}}$ where $\underline{y}[\sigma] \sim P_{\underline{y}}$.

$$E_{\underline{x}}[f(\underline{x})] = \sum_x P_{\underline{x}}(x) f(x) \quad (40.37)$$

$$= \sum_x P_{\underline{y}}(x) \frac{P_{\underline{x}}(x)}{P_{\underline{y}}(x)} f(x) \quad (40.38)$$

$$= E_{\underline{y}}\left[\frac{P_{\underline{x}}(y)}{P_{\underline{y}}(y)} f(y)\right] \quad (40.39)$$

Sampling from $P_{\underline{y}}(y)$ instead of $P_{\underline{x}}(x)$ might reduce (or increase) variance for a particular $f : S_{\underline{x}} \rightarrow \mathbb{R}$.

$$Var_{\underline{x}}[f(x)] = E_{\underline{x}}[(f(x))^2] - (E_{\underline{x}}[f(x)])^2 \quad (40.40)$$

$$Var_{\underline{y}}\left[\frac{P_{\underline{x}}(y)}{P_{\underline{y}}(y)} f(y)\right] = E_{\underline{y}}\left[\left(\frac{P_{\underline{x}}(y)}{P_{\underline{y}}(y)} f(y)\right)^2\right] - (E_{\underline{y}}\left[\frac{P_{\underline{x}}(y)}{P_{\underline{y}}(y)} f(y)\right])^2 \quad (40.41)$$

$$= E_{\underline{x}}\left[\frac{P_{\underline{x}}(x)}{P_{\underline{y}}(x)} (f(x))^2\right] - (E_{\underline{x}}[f(x)])^2 \quad (40.42)$$

Chapter 41

Markov Chains

A Markov Chain is simply a bnet with the graph structure of a chain. For example, Fig.41.1 shows a chain with $n = 4$ nodes.

$$\underline{x}_0 \longrightarrow \underline{x}_1 \longrightarrow \underline{x}_2 \longrightarrow \underline{x}_3$$

Figure 41.1: Markov chain with $n = 4$ nodes.

Because of its graph structure, the TPM of each node only depends on the state of the previous node:

$$P(x_t | (x_a)_{a \neq t}) = P(x_t | x_{t-1}), \quad (41.1)$$

where $(x_a)_{a \neq t}$ are all the nodes except x_t itself and $t = 1, 2, \dots, n - 1$.

If there exists a single TPM $P_{\underline{x}_1 | \underline{x}_0}$ such that

$$P(x_t | x_{t-1}) = P_{\underline{x}_1 | \underline{x}_0}(x_t | x_{t-1}) \quad (41.2)$$

for $t = 1, 2, \dots, n - 1$, then we say that the Markov chain is **time homogeneous**.

Claim 50 (*Data Processing Inequality (DPI)*)

Consider a Markov chain $\underline{x}_0 \rightarrow \underline{x}_1 \cdots \rightarrow \underline{x}_{n-1}$. Suppose $0 \leq a < m < b \leq n - 1$. Then

$$H(\underline{x}_a : \underline{x}_b) \leq \min[H(\underline{x}_a : \underline{x}_m), H(\underline{x}_m : \underline{x}_b)] \quad (41.3)$$

See Ref.[83] for references where the DPI is proven. This inequality confirms our intuitive expectations that the information transmitted (i.e., the mutual information(MI)) from a to b (or vice versa since MI is symmetric) is smaller or equal to the one transmitted from a to m or from m to b because a and b are “farther apart” and “some info can get lost during transmission through the mediator node m ”.

Chapter 42

Mediation Analysis

This chapter is mostly based on Refs.[40, 38] by Pearl.

To fully understand this chapter, the reader should first read Chapter 10 where do and imagine operators are defined.

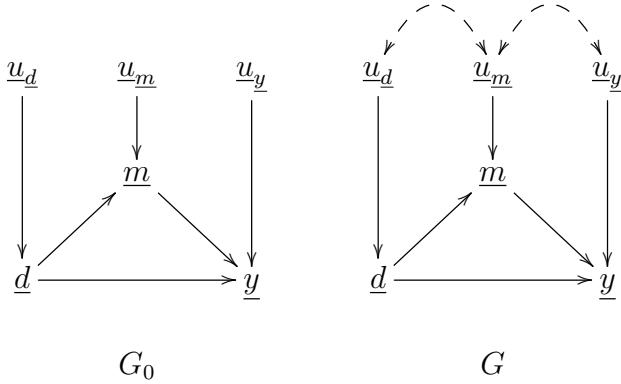


Figure 42.1: DEN bnets G_0 and G are used to discuss mediation analysis. In graph G_0 , the external variables are independent, whereas in graph G they are not.

The term “mediation analysis” (MA) refers to the analysis by Pearl of the DEN bnet G in Fig.42.1. We will discuss MA in terms of DEN bnets, just as Pearl does. However, note that much of what we will say applies also to the general (i.e., not necessarily a DEN) bnet G_{gen} show in Fig. 42.2.

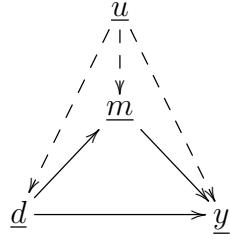
In the DEN bnet G , node \underline{d} influences node \underline{y} both directly and via the mediator node \underline{m} . The structural equations for G are of the form:

$$\underline{d} = \underline{u}_d \tag{42.1a}$$

$$\underline{m} = f_{\underline{m}}(\underline{d}, \underline{u}_m) \tag{42.1b}$$

$$\underline{y} = f_{\underline{y}}(\underline{d}, \underline{m}, \underline{u}_y). \tag{42.1c}$$

Thus,



$$G_{gen}$$

Figure 42.2: General bnet used to discuss mediation analysis. Node \underline{u} is a hidden confounder.

$$\underline{y} = f_{\underline{y}}(\underline{u}_d, f_{\underline{m}}(\underline{u}_d, \underline{u}_m), \underline{u}_y) . \quad (42.2)$$

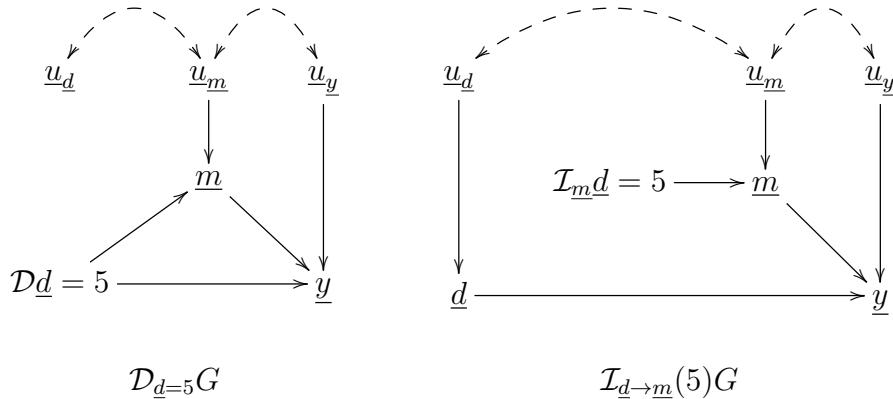


Figure 42.3: Graph G of Fig.42.1 after applying do operator $\mathcal{D}_{\underline{d}=5}$ and imagine operator $\mathcal{I}_{\underline{d} \rightarrow \underline{m}}(5)$.

If we apply $\mathcal{D}_{\underline{d}=5}G$ to Eqs.(42.1), we get

$$\underline{d} = 5 \quad (42.3a)$$

$$\underline{m} = f_{\underline{m}}(\underline{d}, \underline{u}_m) \quad (42.3b)$$

$$\underline{y} = f_{\underline{y}}(\underline{d}, \underline{m}, \underline{u}_y) . \quad (42.3c)$$

Eqs.42.3 are represented graphically in Fig.42.3. We will often denote the random variable \underline{y} in Eqs.(42.3) by the more explicit symbol $\underline{y}_{\mathcal{D}_{\underline{d}=5}G}$. Pearl often refers to this \underline{y} by the less explicit symbol Y_5 or $Y_5(u)$ where $u = (\underline{u}_m, \underline{u}_y) = u_{!d}$.

If we apply $\mathcal{I}_{\underline{d} \rightarrow \underline{m}}(5)G$ to Eqs.(42.1), we get

$$\underline{d} = \underline{u}_d \quad (42.4a)$$

$$\underline{m} = f_{\underline{m}}(5, \underline{u}_m) \quad (42.4b)$$

$$\underline{y} = f_{\underline{y}}(\underline{d}, \underline{m}, \underline{u}_y). \quad (42.4c)$$

Eqs.42.4 are represented graphically in Fig.42.3. We will often denote the random variable \underline{y} in Eqs.(42.4) by the more explicit symbol $\underline{y}_{\mathcal{I}_{\underline{d} \rightarrow \underline{m}}(5)G}$. Pearl often refers to this \underline{y} by the less explicit symbol Y_5 or $Y_5(u)$ where $u = (\underline{u}_d, \underline{u}_m, \underline{u}_y)$.

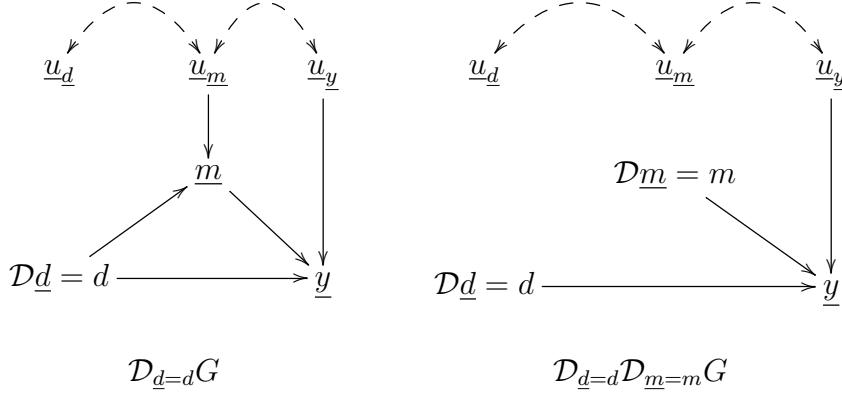


Figure 42.4: Graph G of Fig.42.1 after applying the do operators $\mathcal{D}_{\underline{d}=d}$ and $\mathcal{D}_{\underline{d}=d}\mathcal{D}_{\underline{m}=m}$.

Define

$$\mathcal{Y}_d = E[\underline{y}_{\mathcal{D}_{\underline{d}=d}G}] = \sum_y y P(y | \mathcal{D}\underline{d} = d) \quad (42.5)$$

and

$$\mathcal{Y}_d^m = E[\underline{y}_{\mathcal{D}_{\underline{d}=d}\mathcal{D}_{\underline{m}=m}G}] = \sum_y y P(y | \mathcal{D}\underline{d} = d, \mathcal{D}\underline{m} = m) \quad (42.6)$$

The two DEN diagrams $\mathcal{D}_{\underline{d}=d}G$ and $\mathcal{D}_{\underline{d}=d}\mathcal{D}_{\underline{m}=m}G$ used in the definitions of \mathcal{Y}_d and \mathcal{Y}_d^m are given in Fig.42.4.

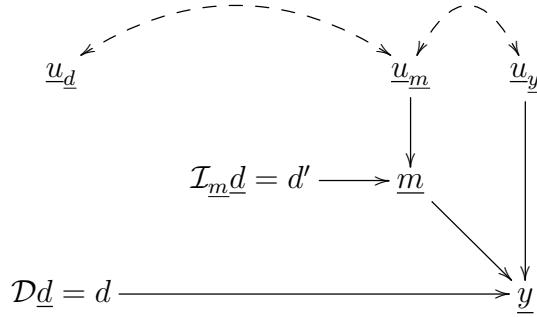
Now define the Total Effect (TE), and the Controlled Direct Effect (CDE) by

$$TE = \mathcal{Y}_1 - \mathcal{Y}_0 \quad (42.7)$$

$$CDE(m) = \mathcal{Y}_1^m - \mathcal{Y}_0^m \quad (42.8)$$

Define

$$\bar{\mathcal{Y}}_d^{d'} = E[\underline{y}_{\mathcal{D}_{\underline{d}=d}\mathcal{I}_{\underline{d} \rightarrow \underline{m}}(d')G}] = \sum_y y P(y | \mathcal{D}\underline{d} = d, \mathcal{I}_m\underline{d} = d') \quad (42.9)$$



$$\mathcal{D}_{\underline{d}=d} \mathcal{I}_{\underline{d} \rightarrow \underline{y}}(d') G$$

Figure 42.5: Graph G of Fig.42.1 after applying the imagine operator \mathcal{I} to arrow $\underline{d} \rightarrow \underline{m}$ and the do operator to node \underline{d} .

Fig.42.5 shows the diagram $\mathcal{D}_{\underline{d}=d} \mathcal{I}_{\underline{d} \rightarrow \underline{y}}(d') G$ used in the definition of $\bar{\mathcal{Y}}_d^{d'}$.

Note that

$$\bar{\mathcal{Y}}_d^{d'} = \sum_m \mathcal{Y}_d^m P(\underline{m} = m | d') \quad (42.10)$$

if there is no arrow $\underline{u}_m \rightarrow \underline{m}$. This expresses $P(y | \mathcal{D}\underline{d} = d, \mathcal{I}\underline{d} = d')$ in terms of $P(y | \mathcal{D}\underline{d} = d, \mathcal{D}\underline{m} = m)$.

Define

$$\bar{\mathcal{Y}}_d^{d'-} = \bar{\mathcal{Y}}_d^{d'} - \bar{\mathcal{Y}}_d^d \quad (42.11)$$

and

$$\bar{\mathcal{Y}}_{d-}^{d'} = \bar{\mathcal{Y}}_d^{d'} - \bar{\mathcal{Y}}_d^d \quad (42.12)$$

Now define the Natural Direct Effect (NDE), and the Natural Indirect Effect (NIE) by

$$NDE_d^{d'} = \bar{\mathcal{Y}}_d^{d'-} \quad (42.13)$$

$$NIE_d^{d'} = \bar{\mathcal{Y}}_{d-}^{d'} . \quad (42.14)$$

Note that

$$NDE_1^0 - NIE_1^0 = -\bar{\mathcal{Y}}_0^0 + \bar{\mathcal{Y}}_1^1 \quad (42.15)$$

$$= TE . \quad (42.16)$$

Linear Case

Consider the LDEN of Fig.42.6. One has

$$\underline{d} = \underline{u}_d \quad (42.17a)$$

$$\underline{m} = \alpha \underline{d} + \underline{u}_m \quad (42.17b)$$

$$\underline{y} = \gamma \underline{d} + \beta \underline{m} + \underline{u}_y . \quad (42.17c)$$

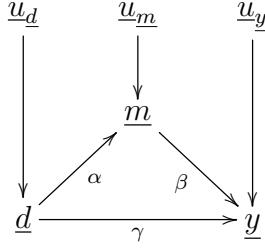


Figure 42.6: LDEN that is used to discuss mediation analysis.

$$\mathcal{Y}_d = (\gamma + \alpha\beta)d \quad (42.18)$$

$$\mathcal{Y}_d^m = \gamma d + \beta m \quad (42.19)$$

$$\bar{\mathcal{Y}}_d^{d'} = \gamma d + \alpha\beta d' \quad (42.20)$$

$$TE = \mathcal{Y}_1 - \mathcal{Y}_0 = \gamma + \alpha\beta \quad (42.21)$$

$$CDE(m) = \mathcal{Y}_1^m - \mathcal{Y}_0^m = \gamma \quad (42.22)$$

$$NDE_1^0 = \bar{\mathcal{Y}}_1^{0-} = \gamma \quad (42.23)$$

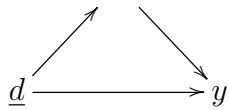
$$NIE_1^0 = \bar{\mathcal{Y}}_{1-}^0 = -\alpha\beta \quad (42.24)$$

As expected, $NDE_1^0 - NIE_1^0 = TE$.

TE and the “controlled effect” $CDE(m)$ contain do operators but no imagine operators so one can do do-intervention experiments to calculate them. On the other hand, the “natural effects” NDE_1^0 and NIE_1^0 contain imagine operators (and therefore counterfactual distributions) so it’s not obvious how to calculate them, or even whether it is possible to do so. The next claim shows how to calculate $\bar{\mathcal{Y}}_d^{d'}$ in certain cases. In technical jargon, the claim gives sufficient conditions for \mathcal{DI} -identifiability of $\bar{\mathcal{Y}}_d^{d'}$. NDE_1^0 and NIE_1^0 can be calculated once $\bar{\mathcal{Y}}_d^{d'}$ is known.

Claim 51 (*Unconfounded Mediation, from Ref.[40]*)

If $\begin{array}{c} m \\ \nearrow \searrow \\ d & \xrightarrow{\hspace{1cm}} & y \end{array}$ then



$$P(y|\mathcal{D}\underline{d} = d, \mathcal{I}\underline{m}d = d') = \sum_m P(y|d, m)P(m|d') \quad (42.25)$$

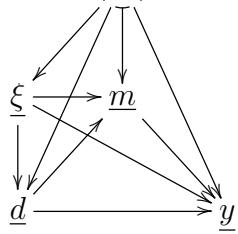
$$\begin{array}{ccc} \mathcal{I}\underline{d} = d' & & \mathcal{I}\underline{d} = d' \longrightarrow \sum m \\ \searrow & & \swarrow \\ \mathcal{D}\underline{d} = d \longrightarrow y & = & \mathcal{D}\underline{d} = d \longrightarrow y \end{array} \quad (42.26)$$

proof: See Claim 31.

QED

Claim 52 (*Mediation with universal prior ξ and universal confounder \underline{u} , from Ref.[40]*)

If $\begin{array}{c} (\underline{u}) \\ \downarrow \\ \xi \end{array}$ then



$$P(y|\mathcal{D}\underline{d} = d, \mathcal{I}_m \underline{d} = d') = \sum_{\xi} \sum_m P(y|d, m, \xi) P(m|d', \xi) P(\xi) \quad (42.27)$$

$$\begin{array}{ccc} \mathcal{I}\underline{d} = d' & & \mathcal{I}\underline{d} = d' \xrightarrow{\sum \xi} \sum m \\ \searrow & & \swarrow \\ \mathcal{D}\underline{d} = d \longrightarrow y & = & \mathcal{D}\underline{d} = d \longrightarrow y \end{array} \quad (42.28)$$

proof: See Claim 32.

QED

Actually,

$$\sum_m P(y|d, m) P(m|d') = \sum_{\xi} \sum_m P(y|d, m, \xi) P(m|d', \xi) P(\xi) \quad (42.29)$$

so the adjustment formulae in Claims 51 and 52 are equivalent, but if the available dataset contains info about ξ , then the adjustment formula that uses that ξ info should be used, as it will be more sensitive to deviations from the DAG model being hypothesized.

Chapter 43

Message Passing (Belief Propagation)

Belief Propagation was first proposed in 1982 Ref.[36] by Judea Pearl to simplify the exact evaluation of the probability of one node conditioned on other nodes of a bnet (exact inference). It gives exact results for trees and polytrees (i.e. bnets with a single connected component and no loops). For bnets with loops, it gives approximate results (loopy belief propagation), and it has been generalized to the junction tree algorithm (see Chapter 35) which can do exact inference for general bnets with loops. The basic idea behind the junction tree algorithm is to eliminate loops by clustering them into single nodes.

In his book Ref.[37], Pearl explains two types of Message Passing (i.e., distributed computing in a bnet). In Chapter 4, he discusses one type of MP which he calls Belief Propagation (BP) or Belief Updating. In Chapter 5, he introduces a second type of MP which he calls Belief Revision, but which I prefer to call Explanation Optimization (EO). This chapter will be devoted to BP only.

This chapter is mostly based on chapter 4 of Ref.[37] by Pearl. Refs.[69], and [30] were also helpful in writing this chapter.

43.1 Distributed Soldier Counting

Consider a group of soldiers marching single file. Fig.43.1 shows several methods by which a member of the group can obtain a count of the soldiers without breaking the line to do global operations. This can be done in a distributed fashion, with every soldier doing only local operations (i.e., each soldier can only send messages to either the soldier in front or the one in back). Such distributed soldier counting is a rudimentary type of BP. In the next section, we will generalize this BP for soldiers to BP for a Markov chain.

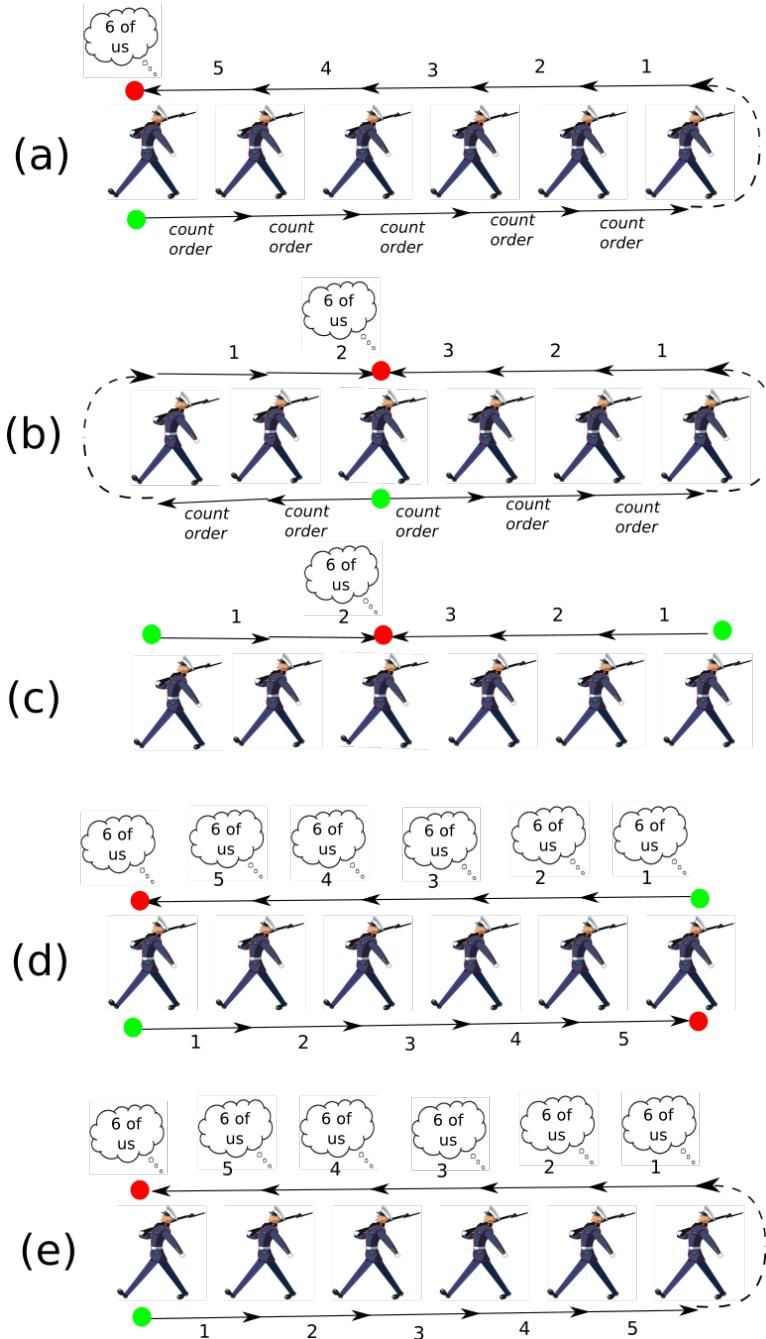


Figure 43.1: Distributed soldier counting (This example comes from Chapter 4 of Ref.[37]). Green dots indicate the beginning and red dots the end of a count. Only first soldier can calculate total count in (a). Only third soldier can calculate total count in (b,c). All soldiers can calculate the total count in (d,e). One starting point in (a,b,e). Two ends as starting points in (c,d).

43.2 Spring Systems

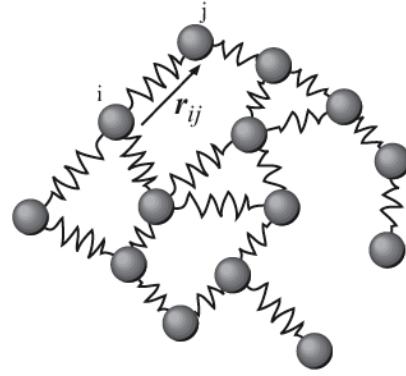


Figure 43.2: Spring system. Point masses connected by springs.

See Ref.[126] for an introduction to spring systems. Ideal springs between the point mass nodes would not be sufficient. One would have to add damping to the springs so as to reach an equilibrium. Time dependent forces (loads) pointing into or out of the page, applied to the point masses, would generate signals that would propagate like BP messages.

43.3 BP for Markov Chains

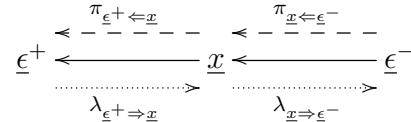


Figure 43.3: 3 node Markov chain $\underline{\epsilon}^+ \leftarrow \underline{x} \leftarrow \underline{\epsilon}^-$. The π messages (probability functions) travel downstream (i.e., they carry info in the direction of the graph arrows, towards the future) and are indicated by a dashed arrow or by a left double arrow \Leftarrow . The λ messages (likelihood functions) travel upstream (i.e., they carry info opposite to direction of the graph arrows, towards the past) and are indicated by a dotted arrow or by a right double arrow \Rightarrow . $\underline{\epsilon}^+$ stands for future evidence and $\underline{\epsilon}^-$ for past evidence.

Consider the 3 node Markov chain $\underline{\epsilon}^+ \leftarrow \underline{x} \leftarrow \underline{\epsilon}^-$ shown in Fig.43.3. Define¹

¹The pattern behind these definitions, in case it eludes you, is as follows: the π 's always carry information about the past and the λ 's about the future. But the past or future of what? Of the

$$\pi_{\underline{\epsilon}^+ \leftarrow \underline{x}}(x) = P(x|\epsilon^-) \text{ (past of } \underline{x}) \quad (43.1)$$

$$\lambda_{\underline{\epsilon}^+ \Rightarrow \underline{x}}(x) = P(\epsilon^+|x) \text{ (future of } \underline{x}) \quad (43.2)$$

$$\pi_{\underline{x} \leftarrow \underline{\epsilon}^-}(\epsilon^-) = P(\epsilon^-) = \delta(\epsilon^-, \epsilon_0^-) \text{ (past of } \underline{\epsilon}^-) \quad (43.3)$$

$$\lambda_{\underline{x} \Rightarrow \underline{\epsilon}^-}(\epsilon^-) = P(\epsilon^+|\epsilon^-) \text{ (future of } \underline{\epsilon}^-) . \quad (43.4)$$

Furthermore, define the Belief BEL in x to be

$$BEL_{\underline{x}}(x) = P(x|\epsilon) , \quad (43.5)$$

where

$$\epsilon = \underline{\epsilon}^+ \cup \underline{\epsilon}^- . \quad (43.6)$$

It follows that

$$BEL_{\underline{x}}(x) = P(x|\epsilon^+, \epsilon^-) = \quad (43.7)$$

$$= \mathcal{N}(!x)P(\epsilon^+, x, \epsilon^-) \quad (43.8)$$

$$= \mathcal{N}(!x)P(\epsilon^+|x)P(x|\epsilon^-) \quad (43.9)$$

$$= \mathcal{N}(!x)\lambda_{\underline{\epsilon}^+ \Rightarrow \underline{x}}(x)\pi_{\underline{\epsilon}^+ \leftarrow \underline{x}}(x) . \quad (43.10)$$

Note that Bayes rule would affirm that²

$$P(x|\epsilon^+) = \mathcal{N}(!x) \underbrace{P(\epsilon^+|x)}_{\lambda_{\underline{\epsilon}^+ \Rightarrow \underline{x}}(x)} P(x) . \quad (43.11)$$

Thus, Eq.(43.10) is like a 2-sided Janus Bayes rule.

Note that the π messages and λ messages propagate independently of each other, via the TPM $P(x|\epsilon^-)$:

$$\underbrace{\pi_{\underline{\epsilon}^+ \leftarrow \underline{x}}(x)}_{P(x|\epsilon_0^-)} = \sum_{\epsilon^-} P(x|\epsilon^-) \underbrace{\pi_{\underline{x} \leftarrow \underline{\epsilon}^-}(\epsilon^-)}_{\delta(\epsilon^-, \epsilon_0^-)} \quad (43.12a)$$

$$\underbrace{\lambda_{\underline{x} \Rightarrow \underline{\epsilon}^-}(\epsilon^-)}_{P(\epsilon^+|\epsilon^-)} = \sum_x P(x|\epsilon^-) \underbrace{\lambda_{\underline{\epsilon}^+ \Rightarrow \underline{x}}(x)}_{P(\epsilon^+|x)} \quad (43.12b)$$

argument of the function. Out of the two random variables in the subscript of the function, the one on the right hand side of the subscript, the one which is adjacent but beneath the argument, is always the argument.

²As usual in this book, $\mathcal{N}(!x)$ means a constant that is independent of x .

Eqs.(43.12) suggest that we define an edge bnet for the π and λ messages (these messages live in the edges between the nodes $\underline{\epsilon}^+, \underline{x}, \underline{\epsilon}^-$). Such an edge bnet, shown in Fig.43.4, is complementary to bnet for the nodes themselves. We will call it the **BP 2-track bnet** for the bnet Fig.43.3, because it has two “tracks”, one for π messages and another for λ ones. The TPMs, printed in blue, for bnet Fig.43.4, are as follows:

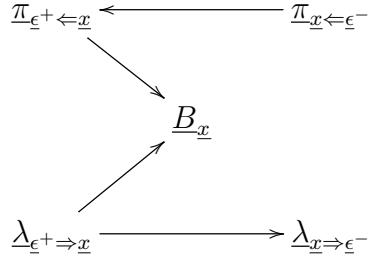


Figure 43.4: BP 2-track bnet for the bnet Fig.43.3.

$$P(\pi_{\underline{x} \leftarrow \underline{\epsilon}^-}) = \prod_{\epsilon^-} \mathbb{1}(\pi_{\underline{x} \leftarrow \underline{\epsilon}^-}(\epsilon^-) = P(\epsilon^-)) \quad (43.13)$$

$$P(\pi_{\underline{\epsilon}^+ \leftarrow \underline{x}} | \pi_{\underline{x} \leftarrow \underline{\epsilon}^-}) = \prod_x \mathbb{1} \left(\pi_{\underline{\epsilon}^+ \leftarrow \underline{x}}(x) = \sum_{\epsilon^-} P(x|\epsilon^-) \pi_{\underline{x} \leftarrow \underline{\epsilon}^-}(\epsilon^-) \right) \quad (43.14)$$

$$P(B_x | \pi_{\underline{\epsilon}^+ \leftarrow \underline{x}}, \lambda_{\underline{\epsilon}^+ \Rightarrow \underline{x}}) = \prod_x \mathbb{1}(B_x(x) = BEL_x(x)) \quad (43.15)$$

$$P(\lambda_{\underline{\epsilon}^+ \Rightarrow \underline{x}}) = \prod_x \mathbb{1}(\lambda_{\underline{\epsilon}^+ \Rightarrow \underline{x}}(x) = P(\epsilon^+|x)) \quad (43.16)$$

$$P(\lambda_{\underline{x} \Rightarrow \underline{\epsilon}^-} | \lambda_{\underline{\epsilon}^+ \Rightarrow \underline{x}}) = \prod_{\epsilon^-} \mathbb{1} \left(\lambda_{\underline{x} \Rightarrow \underline{\epsilon}^-}(\epsilon^-) = \sum_x P(x|\epsilon^-) \lambda_{\underline{\epsilon}^+ \Rightarrow \underline{x}}(x) \right) \quad (43.17)$$

So far in this section, we have considered Markov chains with 3 nodes. Before concluding our discussion of BP for Markov chains, let us consider BP for a slightly longer chain. Let us consider the 5 node Markov chain $\underline{\epsilon}^+ \leftarrow \underline{b} \leftarrow \underline{x} \leftarrow \underline{a} \leftarrow \underline{\epsilon}^-$ shown in Fig.43.5. We have already dealt with the end nodes of a Markov chain in the 3

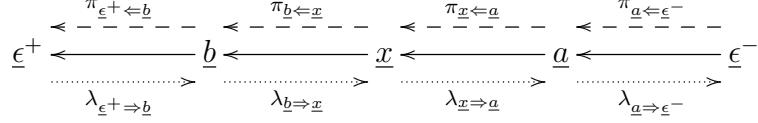


Figure 43.5: 5 node Markov chain

node Markov chain example above, so in the 5 node case, let us focus on the internal (i.e., not at an end) node \underline{x} and its neighbors \underline{a} and \underline{b} . Define

$$\pi_{\underline{b}\leftarrow\underline{x}}(x) = P(x|\epsilon^-) \text{ (past of } \underline{x}) , \quad (43.18)$$

$$\lambda_{\underline{b}\Rightarrow\underline{x}}(x) = P(\epsilon^+|x) \text{ (future of } \underline{x}) , \quad (43.19)$$

$$\pi_{\underline{x}\leftarrow\underline{a}}(a) = P(a|\epsilon^-) \text{ (past of } \underline{a}) \quad (43.20)$$

and

$$\lambda_{\underline{x}\Rightarrow\underline{a}}(a) = P(\epsilon^+|a) \text{ (future of } \underline{a}) . \quad (43.21)$$

Define the Belief BEL in x to be

$$BEL_{\underline{x}}(x) = P(x|\epsilon) , \quad (43.22)$$

where

$$\underline{\epsilon} = \underline{\epsilon}^+ \cup \underline{\epsilon}^- . \quad (43.23)$$

Then

$$BEL_{\underline{x}}(x) = \mathcal{N}(!x)P(\epsilon^+|x)P(x|\epsilon^-) \quad (43.24)$$

$$= \mathcal{N}(!x)\lambda_{\underline{b}\Rightarrow\underline{x}}(x)\pi_{\underline{b}\leftarrow\underline{x}}(x) . \quad (43.25)$$

In analogy with the case of BP for a 3 node Markov chain, we can define the bnet Fig.43.6, which we refer to as the **BP 2-track bnet** for Fig.43.5. The TPMs, printed in blue, for bnet Fig.43.6, are as follows:

$$P(\pi_{\underline{b}\leftarrow\underline{x}}|\pi_{\underline{x}\leftarrow\underline{a}}) = \prod_x \mathbb{1} \left(\pi_{\underline{b}\leftarrow\underline{x}}(x) = \sum_a P(x|a)\pi_{\underline{x}\leftarrow\underline{a}}(a) \right) \quad (43.26)$$

$$P(B_{\underline{x}}|\pi_{\underline{b}\leftarrow\underline{x}}, \lambda_{\underline{b}\Rightarrow\underline{x}}) = \prod_x \mathbb{1} (B_{\underline{x}}(x) = BEL_{\underline{x}}(x)) \quad (43.27)$$

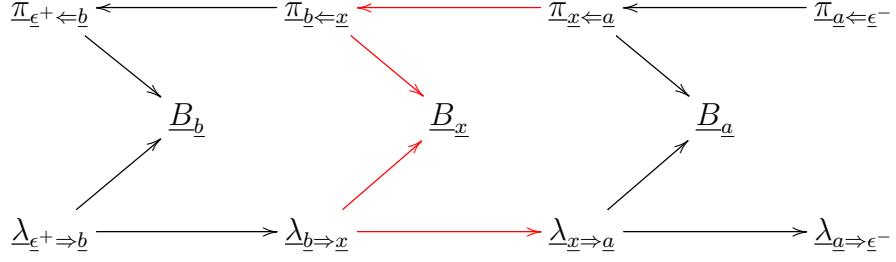


Figure 43.6: BP 2-track bnet for the bnet Fig.43.5.

$$P(\lambda_{\underline{x} \Rightarrow a} | \lambda_{b \Rightarrow \underline{x}}) = \prod_a \mathbb{1} \left(\lambda_{\underline{x} \Rightarrow a}(a) = \sum_x P(x|a) \lambda_{b \Rightarrow \underline{x}}(x) \right) \quad (43.28)$$

Let us represent the Markov chain of Fig.43.5 by $\underline{x}_{nx-1} \leftarrow \dots \leftarrow \underline{x}_2 \leftarrow \underline{x}_1 \leftarrow \underline{x}_0$ where $nx = 5$. For any node \underline{x}_i with parent $\underline{px}_i = \underline{x}_{i-1}$ and child $\underline{cx}_i = \underline{x}_{i+1}$, define the **memory matrix** $\mathcal{M}_{\underline{x}_i}$ for node \underline{x}_i as

$$\mathcal{M}_{\underline{x}_i} = [\mathcal{M}_{\underline{x}_i}^+, \mathcal{M}_{\underline{x}_i}^-], \quad (43.29)$$

where $+$ =future, $-$ =past, and

$$\mathcal{M}_{\underline{x}_i}^+ = \begin{bmatrix} \pi_{\underline{c}\underline{x}_i \leftarrow \underline{x}_i}(\cdot) \\ \lambda_{\underline{c}\underline{x}_i \rightarrow \underline{x}_i}(\cdot) \end{bmatrix}, \quad \mathcal{M}_{\underline{x}_i}^- = \begin{bmatrix} \pi_{\underline{x}_i \leftarrow \underline{p}\underline{x}_i}(\cdot) \\ \lambda_{\underline{x}_i \rightarrow \underline{p}\underline{x}_i}(\cdot) \end{bmatrix}. \quad (43.30)$$

Note that

$$\mathcal{M}_{\underline{x}_i}^- = \mathcal{M}_{\underline{p}\underline{x}_i}^+ \quad (43.31)$$

for all nodes \underline{x}_i . We will refer to Eqs.(43.31) as the **memory overlap conditions**.

We will also use a permuted version of the memory matrix

$$\mathcal{M}'_{\underline{x}_i} = [\mathcal{M}_{\underline{x}_i}^{OUT}, \mathcal{M}_{\underline{x}_i}^{IN}], \quad (43.32)$$

where

$$\mathcal{M}_{\underline{x}_i}^{OUT} = \begin{bmatrix} \pi_{\underline{c}\underline{x}_i \leftarrow \underline{x}_i}(\cdot) \\ \lambda_{\underline{x}_i \rightarrow \underline{p}\underline{x}_i}(\cdot) \end{bmatrix}, \quad \mathcal{M}_{\underline{x}_i}^{IN} = \begin{bmatrix} \pi_{\underline{x}_i \leftarrow \underline{p}\underline{x}_i}(\cdot) \\ \lambda_{\underline{c}\underline{x}_i \rightarrow \underline{x}_i}(\cdot) \end{bmatrix}. \quad (43.33)$$

$$\underline{\mathcal{M}}_{\underline{\epsilon}^+} \leftarrow \underline{\mathcal{M}}_b \leftarrow \underline{\mathcal{M}}_{\underline{x}} \leftarrow \underline{\mathcal{M}}_a \leftarrow \underline{\mathcal{M}}_{\underline{\epsilon}^-}$$

Figure 43.7: BP Memory Bnet for the bnet Fig.43.5.

Unfortunately, 2-track bnets cannot be generalized in any obvious way from Markov chains to more complicated DAGs. An alternative to 2-track bnets that still carries message info in its nodes, are memory bnets. An **BP memory bnet** is a bnet which takes each node of an original bnet and adds a local memory to it. More specifically, it keeps that DAG but replaces each node \underline{x}_i by a memory $\mathcal{M}_{\underline{x}_i}$. Fig.43.7 shows the memory bnet for the bnet Fig.43.5. The TPM, printed in blue, for the node $\mathcal{M}_{\underline{x}}$ of the memory bnet Fig.43.7, is as follows

$$P(\mathcal{M}_{\underline{x}_i} | \mathcal{M}_{n \in nb(\underline{x}_i)}) = AB , \quad (43.34)$$

where

$$A = \mathbb{1}(\mathcal{M}_{\underline{x}_i}^- = \mathcal{M}_{\underline{px}_i}^+) , \quad (43.35)$$

and

$$B = \mathbb{1}(\mathcal{M}_{\underline{x}_i}^{OUT} = \mathcal{C}(\mathcal{M}_{\underline{x}_i}^{IN})) . \quad (43.36)$$

The function \mathcal{C} , which we will call the **BP local computation**, maps $\mathcal{M}_{\underline{x}_i}^{IN}$ into $\mathcal{M}_{\underline{x}_i}^{OUT}$. More explicitly, \mathcal{C} is defined so that

$$B = \underbrace{P(\pi_{b \leftarrow \underline{x}} | \pi_{\underline{x} \leftarrow a})}_{B_\pi} \underbrace{P(\lambda_{\underline{x} \Rightarrow a} | \lambda_{b \Rightarrow \underline{x}})}_{B_\lambda} , \quad (43.37)$$

where B_π and B_λ are given by Eqs.(43.26) and (43.28), respectively.

The BP memory bnet Fig. 43.7 is a deterministic bnet. A deterministic bnet is basically just a coupled system of equations (CSE) for some unknowns x_i . A CSE per se does not include with it a method for solving for the x_i . Such methods are not unique. For example, for the distributed soldier counting problem, the various methods that we described for counting soldiers are just different methods for solving the same CSE. One can describe a method for solving a CSE using a dynamical bnet.³ To solve the CSE represented by the memory bnet Fig.43.7, we will use the dynamical bnet Fig.43.8. Henceforth, we will refer to Fig.43.8 as an **BP dynamical bnet** for Fig.43.7.

Next, we will explain the meaning of Fig.43.8. Fig.43.8 is a step by step recipe (i.e., algorithm) for solving a CSE, where the unknowns are memory matrices. Each step encoded in Fig.43.8 corresponds to a specific message sending event, where the messages are sent along the edges of the Markov chain Fig.43.5. These message sending events are portrayed in chronological order in Fig.43.9. In that figure, π messages are indicated by dashed red arrows, and λ messages by dotted red arrows. These steps, or message sending events, lead to an updating of the memory matrices that we are solving for. Each step propagates information between the memory nodes.

³The term dynamical bnet was used in Chapter 22 to mean a time inhomogeneous Markov chain, but here we are stretching its meaning to include Markov chains that aren't time inhomogeneous.

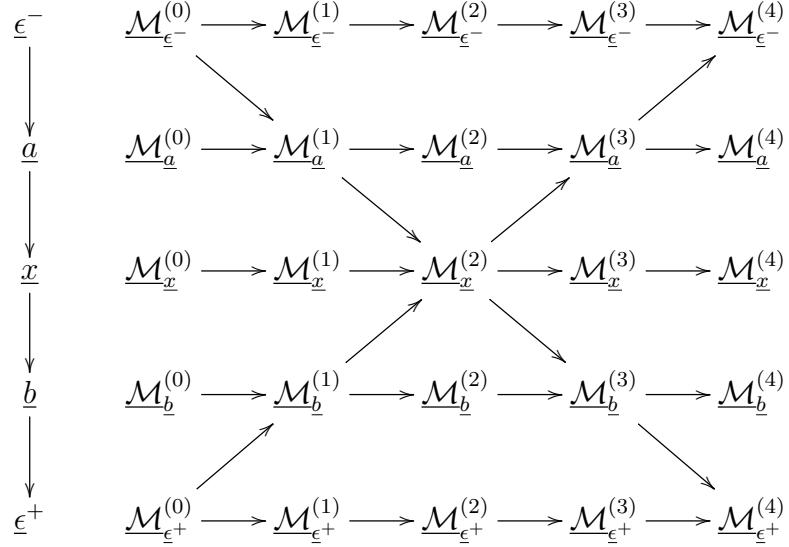


Figure 43.8: BP dynamical bnet for the bnet Fig.43.7.

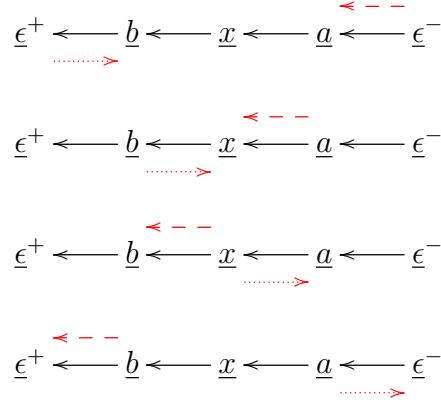


Figure 43.9: Steps encoded in the bnet Fig.43.8. Note the similarity of this figure to Fig.43.1 (d) for soldier counting.

In the usual Pearl BP algo, the evidence nodes initiate the BP chain of message passing events. These events continue until the memory matrices reach an equilibrium and the CSE is solved.

To use bnet Fig.43.8, we need to specify the **initial conditions** (i.e., the value of $\mathcal{M}_{x_i}^{(0)}$ for all i). For that, one can use

$$\pi_{\underline{p}x_0 \Leftarrow x_0}^{(0)} = P(x_0) , \quad (43.38)$$

$$\lambda_{\underline{c}x_i \Rightarrow x_{nx-1}}^{(0)}(x_{nx-1}) = \delta(x_{nx-1}, x'_{nx-1}) . \quad (43.39)$$

All other $\mathcal{M}_{\underline{x}_i}^{(0)}$ entries for all i can be set to 1.

The TPMs, printed in blue, for bnet Fig.43.8, are as follows.

$$P(\mathcal{M}_{\underline{x}_i}^{(t)} | \mathcal{M}_{\underline{n} \in nb(\underline{x}_i)}^{(t-1)}, \mathcal{M}_{\underline{x}_i}^{(t-1)}) = AB, \quad (43.40)$$

where

$$A = \begin{cases} \mathbb{1}(\mathcal{M}_{\underline{x}_i}^{(t)-} = \mathcal{M}_{\underline{px}_i}^{(t-1)+}) & \text{if input from } \underline{px}_i \\ \mathbb{1}(\mathcal{M}_{\underline{x}_i}^{(t)+} = \mathcal{M}_{\underline{cx}_i}^{(t-1)-}) & \text{if input from } \underline{cx}_i \end{cases}, \quad (43.41)$$

and

$$B = \mathbb{1}(\mathcal{M}_{\underline{x}_i}^{(t)OUT} = \mathcal{C}(\mathcal{M}_{\underline{x}_i}^{(t)IN})). \quad (43.42)$$

The function \mathcal{C} , which we will call the **BP local computation**, maps $\mathcal{M}_{\underline{x}_i}^{(t)IN}$ into $\mathcal{M}_{\underline{x}_i}^{(t)OUT}$. More explicitly, \mathcal{C} is defined so that

$$B = B_\pi B_\lambda \quad (43.43)$$

where

$$B_\pi = \prod_x \mathbb{1} \left(\underbrace{\pi_{\underline{b} \Leftarrow \underline{x}}^{(t)}(x)}_{OUT} = \sum_a P(x|a) \underbrace{\pi_{\underline{x} \Leftarrow \underline{a}}^{(t)}(a)}_{IN} \right) \quad (43.44)$$

and

$$B_\lambda = \prod_a \mathbb{1} \left(\underbrace{\lambda_{\underline{x} \Rightarrow \underline{a}}^{(t)}(a)}_{OUT} = \sum_x P(x|a) \underbrace{\lambda_{\underline{b} \Rightarrow \underline{x}}^{(t)}(x)}_{IN} \right). \quad (43.45)$$

The basic idea behind Eq.(43.42), which we will call the **memory updating equation**, is simple: the memory overlap conditions translate the information from time $t - 1$ to t , and then the local computation translates IN to OUT at fixed time t .

43.4 BP Algorithm for Polytrees

Consider Fig.43.10, which illustrates a bnet node \underline{x} receiving and sending messages to its neighbors. The π messages (probability functions) travel downstream (i.e., they carry info in the direction of the graph arrows, towards the future) and are indicated by a dashed arrow or by a left double arrow \Leftarrow . The λ messages (likelihood functions) travel upstream (i.e., they carry info opposite to direction of the graph

arrows, towards the past) and are indicated by a dotted arrow or by a right double arrow \Rightarrow .

Note that argument arg of the $\pi(arg)$ and $\lambda(arg)$ functions is always the same as the letter in the subscript that is closest to the argument.

Note that in Fig.43.10, we indicate messages that travel “downstream” (resp., “upstream”), by arrows with dashed (resp., dotted) lines as shafts. Mnemonic: think of the shaft as a velocity vector field for the message. You travel faster when you swim downstream as opposed to upstream.

$$pa(\underline{x}) = \text{parents of node } \underline{x}$$

$$ch(\underline{x}) = \text{children of node } \underline{x}$$

$$nb(\underline{x}) = pa(\underline{x}) \cup ch(\underline{x}) = \text{neighbors of node } \underline{x}$$

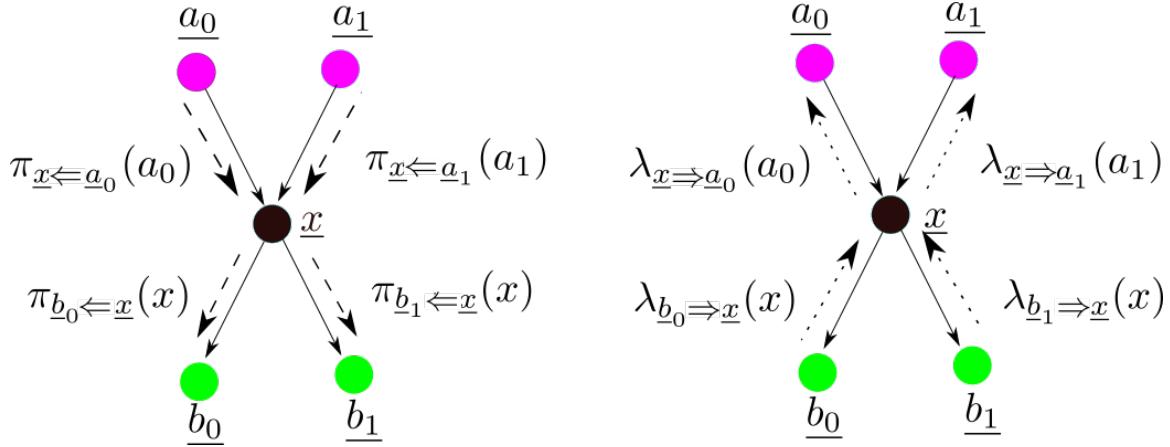


Figure 43.10: Node \underline{x} receiving and sending messages to its neighbors. (neighbors=parents and children).

We define a **memory matrix** $\mathcal{M}_{\underline{x}}$ for node \underline{x} as

$$\mathcal{M}_{\underline{x}} = [\mathcal{M}_{\underline{x}}^+, \mathcal{M}_{\underline{x}}^-] , \quad (43.46)$$

where $+$ =future, $-$ =past, and

$$\mathcal{M}_{\underline{x}}^+ = [\pi_{\underline{b} \leftarrow \underline{x}}(\cdot) \quad \lambda_{\underline{b} \Rightarrow \underline{x}}(\cdot)]_{\underline{b} \in ch(\underline{x})} = [\mathcal{M}_{\underline{b}, \underline{x}}^+]_{\underline{b} \in ch(\underline{x})} , \quad (43.47)$$

$$\mathcal{M}_{\underline{x}}^- = \left[\begin{array}{c} \pi_{\underline{x} \leftarrow \underline{a}}(\cdot) \\ \lambda_{\underline{x} \Rightarrow \underline{a}}(\cdot) \end{array} \right]_{\underline{a} \in pa(\underline{x})} = [\mathcal{M}_{\underline{x}, \underline{a}}^-]_{\underline{a} \in pa(\underline{x})} . \quad (43.48)$$

Note that

$$\mathcal{M}_{\underline{x}, \underline{a}}^- = \mathcal{M}_{\underline{a}, \underline{x}}^+ \quad (43.49)$$

for every arrow $\underline{x} \leftarrow \underline{a}$. We will refer to Eqs.(43.49) as the **memory overlap conditions**.

We will also use a permuted version of the memory matrix

$$\mathcal{M}'_{\underline{x}} = [\mathcal{M}_{\underline{x}}^{OUT}, \mathcal{M}_{\underline{x}}^{IN}] , \quad (43.50)$$

where

$$\mathcal{M}_{\underline{x}}^{OUT} = \left(\begin{array}{c} [\pi_{b \leftarrow \underline{x}}(\cdot)]_{b \in ch(\underline{x})} \\ [\lambda_{\underline{x} \Rightarrow a}(\cdot)]_{a \in pa(\underline{x})} \end{array} \right) = [\mathcal{M}_{\underline{x}, \underline{n}}^{OUT}]_{\underline{n} \in nb(\underline{x})} , \quad (43.51)$$

$$\mathcal{M}_{\underline{x}}^{IN} = \left(\begin{array}{c} [\pi_{x \leftarrow a}(\cdot)]_{a \in pa(\underline{x})} \\ [\lambda_{b \rightarrow \underline{x}}(\cdot)]_{b \in ch(\underline{x})} \end{array} \right) = [\mathcal{M}_{\underline{x}, \underline{n}}^{IN}]_{\underline{n} \in nb(\underline{x})} . \quad (43.52)$$

For times $t = 0, 1, \dots, T - 1$, we calculate $\mathcal{M}_{\underline{x}}^{(t)}$ in two steps: first we calculate $\mathcal{M}_{\underline{x}}^{(t)IN}$ from earlier memories at time $t - 1$, then we calculate $\mathcal{M}_{\underline{x}}^{(t)OUT}$:

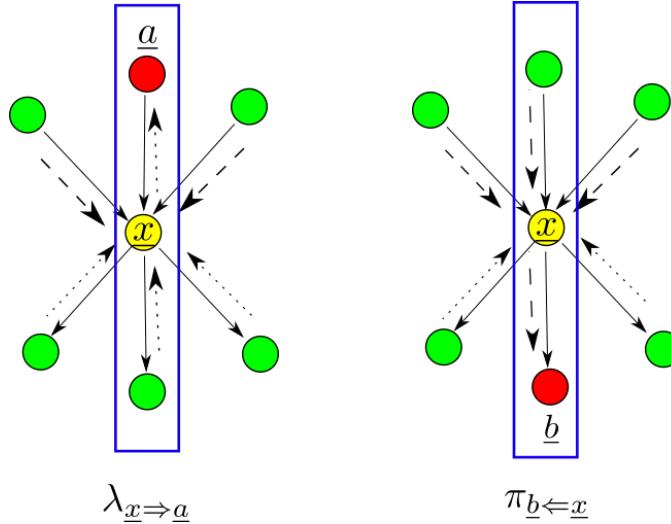


Figure 43.11: Subgraph of a bnet showing two cases (RULE 1 and RULE 2) of message info flow. The yellow node is a gossip monger. It receives messages from all the green nodes, and then it relays a joint message to the red node. Union of green nodes and the red node = full neighborhood of yellow node. There are two possible cases: the red node is either a parent or a child of the yellow one. As usual, we use arrows with dashed (resp., dotted) shafts for downstream (resp., upstream) messages. Blue boxes indicate Markov chain case.

An **evidence node** is a node whose TPM is a delta function set to a particular state of the node. We will assume, without loss of generality, that all evidence nodes are leaf nodes. If that is not the case, any evidence node \underline{e} that is not a leaf node, can be given a new companion leaf node \underline{l} connected to \underline{e} by an arrow $\underline{l} \leftarrow \underline{e}$, and such that \underline{l} has a delta function TPM.

1. Calculating $\mathcal{M}_{\underline{x}}^{(t)IN}$ from signals received from $\underline{n} \in nb(\underline{x})$, sent at earlier time $t - 1$:

Set

$$\mathcal{M}_{\underline{x}, \underline{a}}^{(t)-} |_{\pi} = \mathcal{M}_{\underline{a}, \underline{x}}^{(t-1)+} |_{\pi} , \quad (43.53)$$

for all $\underline{a} \in pa(\underline{x})$, and

$$\mathcal{M}_{\underline{b}, \underline{x}}^{(t)+} |_{\lambda} = \mathcal{M}_{\underline{x}, \underline{b}}^{(t-1)-} |_{\lambda} , \quad (43.54)$$

for all $\underline{b} \in ch(\underline{x})$. By $X|_{\lambda}$ (resp., $X|_{\pi}$) we mean the λ (resp., π) component of X .

2. Calculating $\mathcal{M}_{\underline{x}}^{(t)OUT}$ from already calculated $\mathcal{M}_{\underline{x}}^{(t)IN}$:

Let $\underline{a}^{na} = (\underline{a}_i)_{i=0,1,\dots,na-1}$ denote the parents of \underline{x} and $\underline{b}^{nb} = (\underline{b}_i)_{i=0,1,\dots,nb-1}$ its children.

Define

$$\pi_{\underline{x}}(x) = \sum_{a^{na}} P(x|a^{na}) \prod_i \pi_{\underline{x} \leftarrow \underline{a}_i}(a_i) \quad (43.55)$$

$$= E_{\underline{a}^{na}}[P(x|a^{na})] \quad (43.56)$$

(boundary case: if \underline{x} is a root node, use $\pi_{\underline{x}}(x) = P(x)$.) and

$$\lambda_{\underline{x}}(x) = \prod_i \lambda_{\underline{b}_i \Rightarrow \underline{x}}(x) . \quad (43.57)$$

(boundary case: if \underline{x} is a leaf node, use $\lambda_{\underline{x}}(x) = 1$.)

- RULE 1: (red parent)

From the $\lambda_{\underline{x} \Rightarrow \underline{a}}$ panel of Fig.43.11, we get

$$\underbrace{\lambda_{\underline{x} \Rightarrow \underline{a}_i}(a_i)}_{OUT} = \mathcal{N}(!a_i) \sum_x \left[\underbrace{\lambda_{\underline{x}}(x)}_{IN} \sum_{(a_k)_{k \neq i}} \left(P(x|a^{na}) \prod_{k \neq i} \underbrace{\pi_{\underline{x} \leftarrow \underline{a}_k}(a_k)}_{IN} \right) \right] \quad (43.58)$$

$$= \mathcal{N}(!a_i) \sum_x [\lambda_{\underline{x}}(x) E_{(a_k)_{k \neq i}}[P(x|a^{na})]] \quad (43.59)$$

$$= \mathcal{N}(!a_i) E_{(a_k)_{k \neq i}} E_{\underline{x}|a^{na}} \lambda_{\underline{x}}(x) \quad (43.60)$$

(boundary case: if \underline{x} is a root node, use $\lambda_{\underline{x} \Rightarrow \underline{a}_i}(a_i) = \mathcal{N}(!a_i)$.)

- RULE 2: (red child)

From the $\pi_{\underline{b} \leftarrow \underline{x}}$ panel of Fig.43.11, we get

$$\underbrace{\pi_{\underline{b}_i \leftarrow \underline{x}}(x)}_{OUT} = \mathcal{N}(!x) \underbrace{\pi_{\underline{x}}(x)}_{IN} \prod_{k \neq i} \underbrace{\lambda_{\underline{b}_k \Rightarrow \underline{x}}(x)}_{IN} \quad (43.61)$$

(boundary case: if \underline{x} is a leaf node, use $\pi_{\underline{b}_i \leftarrow \underline{x}}(x) = \mathcal{N}(!x) \pi_{\underline{x}}(x)$.)

In the above equations, if the range set of a product is empty, then define the product as 1; i.e., $\prod_{k \in \emptyset} F(k) = 1$.

Claim: Define

$$BEL^{(t)}(\underline{x}) = \mathcal{N}(!\underline{x}) \lambda_{\underline{x}}^{(t)}(\underline{x}) \pi_{\underline{x}}^{(t)}(\underline{x}) . \quad (43.62)$$

Then

$$\lim_{t \rightarrow \infty} BEL^{(t)}(\underline{x}) = P(\underline{x}|\epsilon) . \quad (43.63)$$

This says that the belief in $\underline{x} = \underline{x}$ converges to $P(\underline{x}|\epsilon)$ and it equals the product of messages received from all parents and children of $\underline{x} = \underline{x}$.

43.4.1 How BP algo for polytrees reduces to the BP algo for Markov chains

It is instructive to see how the BP algo for polytrees reduces to BP algo for Markov chains.

For a Markov chain, node \underline{x} has a single parent (i.e., ancestor) \underline{a} and a single child \underline{b} .

Therefore, Eqs.(43.55) and (43.57) reduce to

$$\pi_{\underline{x}}(\underline{x}) = \sum_a P(\underline{x}|a) \pi_{\underline{x} \leftarrow \underline{a}}(a) \quad (43.64)$$

and

$$\lambda_{\underline{x}}(\underline{x}) = \lambda_{\underline{b} \Rightarrow \underline{x}}(\underline{x}) . \quad (43.65)$$

RULE 1 given by Eq.(43.58) reduces to

$$\lambda_{\underline{x} \Rightarrow \underline{a}}(a) = \mathcal{N}(!a) \sum_x \lambda_{\underline{x}}(x) P(x|a) \quad (43.66)$$

$$= \mathcal{N}(!a) \sum_x \lambda_{\underline{b} \Rightarrow \underline{x}}(x) P(x|a) \quad (43.67)$$

RULE 2 given by Eq.(43.61) reduces to

$$\pi_{\underline{b} \leftarrow \underline{x}}(\underline{x}) = \mathcal{N}(!\underline{x}) \pi_{\underline{x}}(\underline{x}) \quad (43.68)$$

$$= \sum_a P(\underline{x}|a) \pi_{\underline{x} \leftarrow \underline{a}}(a) . \quad (43.69)$$

43.5 Derivation of BP Algorithm for Polytrees

This derivation is taken from the 1988 book Ref.[37] by Judea Pearl, where it is presented very lucidly. We only made some minor changes in notation.

Notation

The BP algorithm yields an expansion for $P(x|\epsilon)$.

\underline{x} = the focus node, arbitrary node of bnet that we are focusing on to calculate its $P(x|\epsilon)$.

$(a_i)_{i=0,1,\dots,na-1}$. = parent nodes (mnemonic: a=ancestor) of \underline{x}

$(b_i)_{i=0,1,\dots,nb-1}$. = children nodes of \underline{x} .

$\underline{\epsilon}$ = set of nodes for which there is evidence; that is, $\underline{\epsilon} = \epsilon$, so the state of these nodes is fixed.

$$\epsilon_{\underline{x}}^- = \epsilon \cap an(\underline{x}) \text{ (evidence in past of } \underline{x})^4$$

$$\epsilon_{xa_i}^- = \epsilon_{\underline{x}}^- \cap an(a_i).$$

$$\text{Note that } \epsilon_{\underline{x}}^- = \cup_i \epsilon_{xa_i}^-$$

$$\epsilon_{\underline{x}}^+ = \epsilon \cap [de(\underline{x}) \cup \underline{x}] \text{ (evidence in future of } \underline{x})$$

$$\epsilon_{xb_i}^+ = \epsilon_{\underline{x}}^+ \cap [de(b_i) \cup b_i].$$

$$\text{Note that } \epsilon_{\underline{x}}^+ = \cup_i \epsilon_{xb_i}^+$$

$$\text{Note that } \underline{\epsilon} = \epsilon_{\underline{x}}^+ \cup \epsilon_{\underline{x}}^-$$

$$\pi_{\underline{x}}(x) = P(x|\epsilon_{\underline{x}}^-) \quad (43.70)$$

$$\pi_{\underline{x} \leftarrow a_i}(a_i) = P(a_i|\epsilon_{xa_i}^-) \quad (43.71)$$

$$\pi_{\underline{b}_i \leftarrow \underline{x}}(x) = P(x|\epsilon_{xb_i}^-) \quad (43.72)$$

$$\lambda_{\underline{x}}(x) = P(\epsilon_{\underline{x}}^+|x) \quad (43.73)$$

$$\lambda_{b_i \Rightarrow \underline{x}}(x) = P(\epsilon_{xb_i}^+|x) \quad (43.74)$$

$$\lambda_{\underline{x} \Rightarrow a_i}(a_i) = P(\epsilon_{xa_i}^+|a_i) \quad (43.75)$$

Expansions of $\lambda_{\underline{x}}(x)$ and $\pi_{\underline{x}}(x)$ into products of single node messages.

⁴Careful: Chapter 4 of Ref.[37] uses – indicate the future and + to indicate the past. This is the opposite of our notation.

$$\underbrace{P(x|\epsilon_{\underline{x}}^-)}_{\pi_{\underline{x}}(x)} = P(x| \cup_i \epsilon_{\underline{x}a_i}^-) \quad (43.76)$$

$$= \sum_{a^{na}} P(x|a^{na}) P(a^{na}| \cup_i \epsilon_{\underline{x}a_i}^-) \quad (43.77)$$

$$= \sum_{a^{na}} P(x|a^{na}) \prod_i \underbrace{P(a_i|\epsilon_{\underline{x}a_i}^-)}_{\pi_{\underline{x} \leftarrow a_i}(a_i)} \quad (43.78)$$

$$\underbrace{P(\epsilon_{\underline{x}}^+|x)}_{\lambda_{\underline{x}}(x)} = \prod_i \underbrace{P(\epsilon_{\underline{x}b_i}^+|x)}_{\lambda_{\underline{b}_i \Rightarrow \underline{x}}(x)} \quad (43.79)$$

Note that past and future evidences $\epsilon_{\underline{x}}^-$ and $\epsilon_{\underline{x}}^+$ that are causally connected to \underline{x} are conditionally independent at fixed \underline{x} :

$$P(\epsilon_{\underline{x}}^+, \epsilon_{\underline{x}}^-|x) = P(\epsilon_{\underline{x}}^+|x)P(\epsilon_{\underline{x}}^-|x). \quad (43.80)$$

This observation is key to the proof of the following claim:

Claim 53

$$P(x|\epsilon_{\underline{x}}^+, \epsilon_{\underline{x}}^-) = P(\epsilon_{\underline{x}}^+|x)P(x|\epsilon_{\underline{x}}^-) \frac{1}{P(\epsilon_{\underline{x}}^+|\epsilon_{\underline{x}}^-)} \quad (43.81)$$

$$= \mathcal{N}(!x)P(\epsilon_{\underline{x}}^+|x)P(x|\epsilon_{\underline{x}}^-) \quad (43.82)$$

$$= \mathcal{N}(!x) (\epsilon_{\underline{x}}^+ \leftarrow x \leftarrow \epsilon_{\underline{x}}^-) \quad (43.83)$$

$$= \mathcal{N}(!x)\lambda_{\underline{x}}(x)\pi_{\underline{x}}(x) \quad (43.84)$$

proof:

$$P(x|\epsilon_{\underline{x}}^+, \epsilon_{\underline{x}}^-) = P(\epsilon_{\underline{x}}^+, \epsilon_{\underline{x}}^-|x) \frac{P(x)}{P(\epsilon_{\underline{x}}^+, \epsilon_{\underline{x}}^-)} \quad (43.85)$$

$$= P(\epsilon_{\underline{x}}^+|x)P(\epsilon_{\underline{x}}^-|x) \frac{P(x)}{P(\epsilon_{\underline{x}}^+, \epsilon_{\underline{x}}^-)} \quad (43.86)$$

$$= P(\epsilon_{\underline{x}}^+|x)P(x|\epsilon_{\underline{x}}^-) \frac{P(\epsilon_{\underline{x}}^-)}{P(\epsilon_{\underline{x}}^+, \epsilon_{\underline{x}}^-)} \quad (43.87)$$

$$= P(\epsilon_{\underline{x}}^+|x)P(x|\epsilon_{\underline{x}}^-) \frac{1}{P(\epsilon_{\underline{x}}^+|\epsilon_{\underline{x}}^-)} \quad (43.88)$$

QED

Next we prove BP rules 1 and 2.

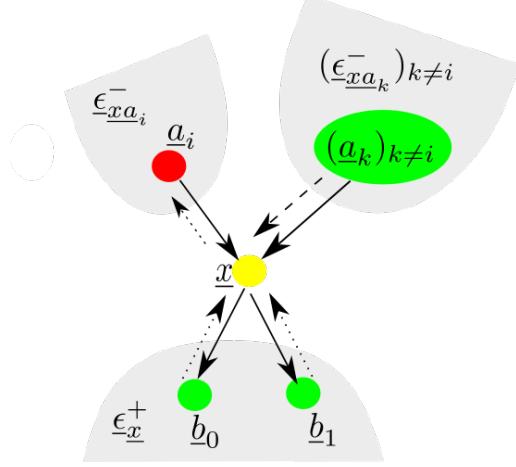


Figure 43.12: This figure is used in the derivation of the BP RULE 1.

- **RULE 1 (red parent)**

Note that

$$\epsilon_{\underline{x}}^+ \cup \cup_{k \neq i} \epsilon_{\underline{x}a_k}^- = (\epsilon_{\underline{x}}^+ \cup \epsilon_{\underline{x}}^-) - \epsilon_{\underline{x}a_i}^- \quad (43.89)$$

$$= \epsilon_{\underline{x}a_i}^+ \quad (43.90)$$

Let $y = (a_k)_{k \neq i}$ and $\epsilon_{\underline{y}}^- = (\epsilon_{\underline{x}a_k}^-)_{k \neq i}$.

$$\underbrace{P(\epsilon_{\underline{x}a_i}^+ | a_i)}_{\lambda_{\underline{x} \Rightarrow \underline{a}_i}(a_i)} = P(\epsilon_{\underline{x}}^+, \epsilon_{\underline{y}}^- | a_i) \quad (43.91)$$

$$= \sum_x \sum_y P(\epsilon_{\underline{x}}^+, \epsilon_{\underline{y}}^- | x, y) P(x, y | a_i) \quad (43.92)$$

$$= \sum_x \sum_y P(\epsilon_{\underline{x}}^+ | x) P(\epsilon_{\underline{y}}^- | y) P(x | y, a_i) P(y | a_i) \quad (43.93)$$

$$= P(\epsilon_{\underline{y}}^-) \sum_x \sum_y P(\epsilon_{\underline{x}}^+ | x) \frac{P(y | \epsilon_{\underline{y}}^-)}{P(y)} P(x | y, a_i) \underbrace{P(y | a_i)}_{=P(y)} \quad (43.94)$$

$$= \mathcal{N}(!a_i) \sum_x \sum_y P(\epsilon_{\underline{x}}^+ | x) P(x | \underbrace{y, a_i}_{a^{na}}) P(y | \epsilon_{\underline{y}}^-) \quad (43.95)$$

$$= \mathcal{N}(!a_i) \sum_x \underbrace{P(\epsilon_{\underline{x}}^+ | x)}_{\lambda_{\underline{x}}(x)} \sum_{(a_k)_{k \neq i}} P(x | a^{na}) \prod_{k \neq i} \underbrace{P(a_k | \epsilon_{\underline{x}a_k}^-)}_{\pi_{\underline{x} \leftarrow \underline{a}_k}(a_k)} \quad (43.96)$$

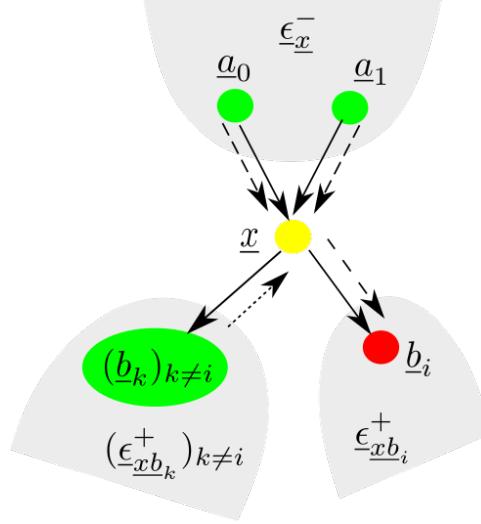


Figure 43.13: This figure is used in the derivation of the BP RULE 2.

- **RULE 2 (red child)**

Note that

$$(\cup_{k \neq i} \epsilon_{\underline{x}b_k}^+) \cup \epsilon_{\underline{x}}^- = (\epsilon_{\underline{x}}^+ \cup \epsilon_{\underline{x}}^-) - \epsilon_{\underline{x}b_i}^+ \quad (43.97)$$

$$= \epsilon_{\underline{x}b_i}^- \quad (43.98)$$

$$\underbrace{P(x|\epsilon_{\underline{x}b_i}^-)}_{\pi_{\underline{b}_i \Leftarrow \underline{x}}(x)} = P(x|(\epsilon_{\underline{x}b_k}^+}_{k \neq i})_{k \neq i}, \epsilon_{\underline{x}}^-) \quad (43.99)$$

$$= \mathcal{N}(!x) P((\epsilon_{\underline{x}b_k}^+}_{k \neq i}|x) P(x|\epsilon_{\underline{x}}^-) \quad (43.100)$$

$$= \mathcal{N}(!x) \left(\prod_{k \neq i} \underbrace{P(\epsilon_{\underline{x}b_k}^+|x)}_{\lambda_{\underline{b}_k \Rightarrow \underline{x}}(x)} \right) \underbrace{P(x|\epsilon_{\underline{x}}^-)}_{\pi_{\underline{x}}(x)} \quad (43.101)$$

43.6 Example of BP algo for a Tree

In this section, we describe how to apply the BP algo to the tree bnet Fig.43.14. In Fig.43.14, if we replace each integer i by the random variable \underline{A}_i , we get an **original bnet**, and if we replace each i by $\underline{\mathcal{M}}_{\underline{A}_i}$, we get the **BP memory bnet** of the original bnet. In Fig.43.14, the magenta nodes are evidence nodes and the green ones aren't.

We want to solve for the memory matrices of the memory bnet. To do so, we use the **BP dynamical bnet** Fig.43.15. The steps encoded in the dynamical bnet are shown in Fig.43.16. Fig.43.16 has frames in chronological order, showing the direction of travel of the $\pi\&\lambda$ information. This sequence of frames also indicates the order in which we solve for the entries of the memory matrices. The information first emanates from the evidence nodes. It propagates generally upstream, although some nodes can generate downstream flow. Some of the info reaches the root node and is reflected there. The root node is the only one that is capable of reflection (i.e., instant output along an arrow, in response to input along that arrow). Eventually, all info reaches the leaf nodes via downstream propagation and is absorbed there.

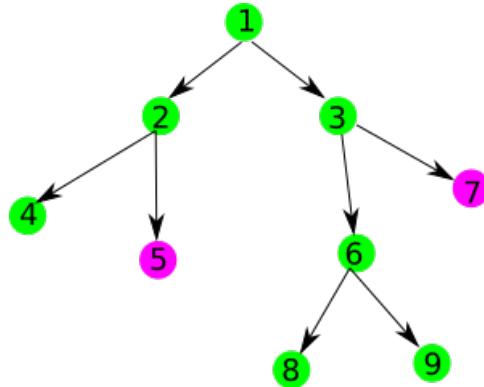


Figure 43.14: Example tree bnet used to illustrate BP.

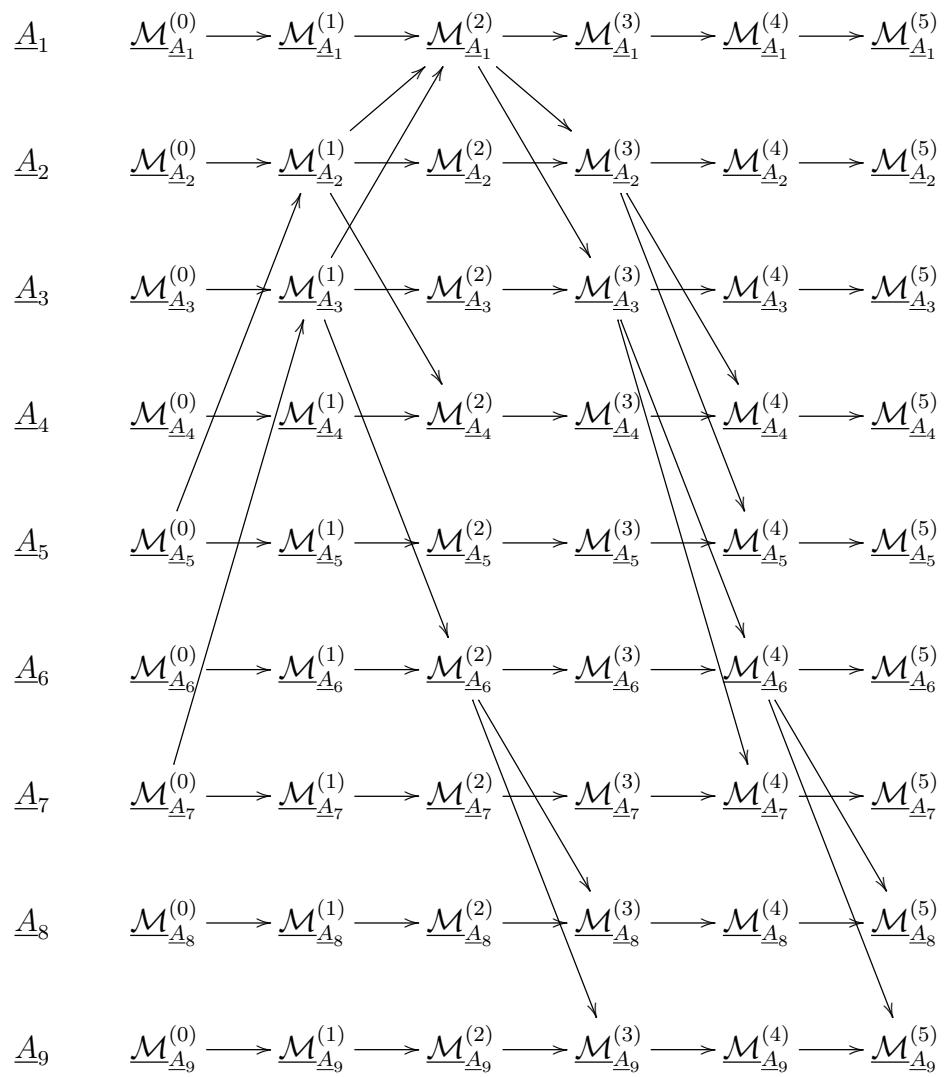


Figure 43.15: BP dynamical bnet for the bnet Fig.43.14.

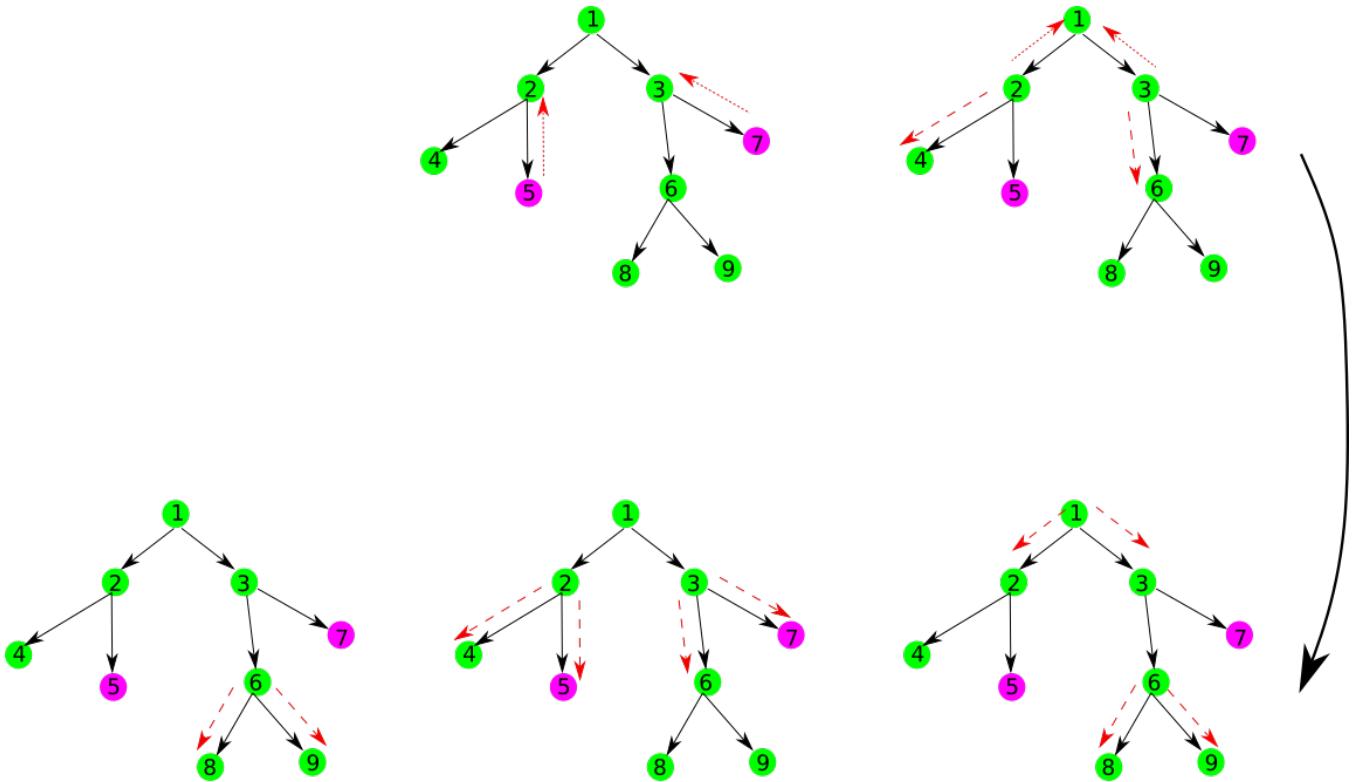


Figure 43.16: Steps encoded in the bnet Fig.43.15.

43.7 Bipartite bnets

By a **bipartite bnet** we will mean a bnet in which all nodes are either root nodes (parentless) or leaf nodes (childless). BP simplifies when dealing with bipartite bnets. In this section, we will explain how it simplifies. But before doing so, let us explain how the following two types of diagrams can be replaced by equivalent bipartite bnets:

- Factor Graphs
- Tree bnets

Consider a product $g = \prod_{\alpha} f_{\alpha}$ of scalar functions $f_{\alpha} : S_{\underline{x}_0} \times S_{\underline{x}_1} \times \dots \times S_{\underline{x}_{n_x-1}} \rightarrow \mathbb{R}$ for $\alpha = 0, 1, \dots, n_f - 1$. For instance, consider $g : S_{\underline{x}_0} \times S_{\underline{x}_1} \times S_{\underline{x}_2} \rightarrow \mathbb{R}$ defined by:

$$g(x_0, x_1, x_2) = f_0(x_0)f_1(x_0, x_1)f_2(x_0, x_1, x_2)f_3(x_1, x_2). \quad (43.102)$$

The **factor graph** for this function g is given by Fig.43.17.

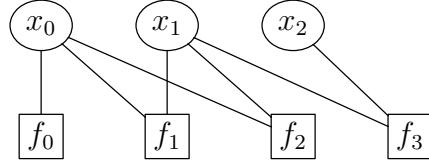


Figure 43.17: Factor graph for function g defined by Eq.(43.102).

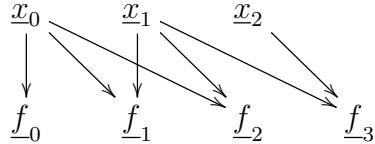


Figure 43.18: Bipartite bnet corresponding to factor graph Fig.43.17.

One can map any factor graph (the “source”) to a special bipartite bnet (the “image”), as follows. Replace each x_i by $\underline{x}_i \in S_{\underline{x}_i}$ for $i = 0, 1, \dots, n_x - 1$ and each f_{α} by \underline{f}_{α} for $\alpha = 0, 1, \dots, n_f - 1$. Then replace the connections (edges) of the factor graph by arrows from \underline{x}_i to \underline{f}_{α} . For example, Fig.43.18 is the image bipartite bnet of the source factor graph Fig.43.17.

Let $\underline{x}^{nx} = (x_0, \underline{x}_1, \dots, \underline{x}_{nx-1})$ and $\underline{f}^{nf} = (\underline{f}_0, \underline{f}_1, \dots, \underline{f}_{nf-1})$. Let⁵ $f_\alpha \in \{0, 1\}$ for all α , and $y_\alpha = f_\alpha(x_{nb(\underline{f}_\alpha)})$. Here we are using $nb(\underline{f}_\alpha)$ to denote the neighborhood of node \underline{f}_α in the image bipartite bnet, and we are using x_S to denote $(x_i)_{i \in S}$. Without loss of generality, we will assume that $y_\alpha \in [0, 1]$ for all α . Then we define the TPMs, printed in blue, for the image bipartite bnet, as follows.

$$P(f_\alpha | x_{nb(\underline{f}_\alpha)}) = y_\alpha \delta(f_\alpha, 1) + [1 - y_\alpha] \delta(f_\alpha, 0) \quad (43.103)$$

for $\alpha = 0, 1, \dots, nf - 1$ and

$$P_{\underline{x}_i}(x_i) = \text{arbitrary prior} \quad (43.104)$$

for $i = 0, 1, \dots, nx - 1$.

Note that

$$P(f^{nf} = 1^{nf} | \underline{x}^{nx}) = \prod_\alpha f_\alpha(x_{nb(\underline{f}_\alpha)}) . \quad (43.105)$$

A **tree bnet** is a bnet for which all nodes have exactly one parent except for the apex root node which has none. A tree bnet is very much like the filing system in a computer.

One can map a tree bnet (the “source”) into an equivalent bipartite bnet (the “image”) as follows. Replace each arrow

$$\underline{x} \longrightarrow \underline{y} \quad (43.106)$$

of the tree bnet by

$$\underline{x} \longrightarrow P_{\underline{y}|\underline{x}} \longleftarrow \underline{y} . \quad (43.107)$$

For example, the tree bnet Fig.43.19 has the image bipartite bnet given by Fig.43.20. The bnet Fig.43.21 is just a different way of drawing the bnet Fig.43.20.

The TPMs, printed in blue, for the image bipartite bnet Fig.43.20, are as follows. We express the TPMs of the image bnet in terms of the TPMs of the source bnet Fig.43.19. Let

$$P(P_{\underline{y}|\underline{x}} | x, y) = P_{\underline{y}|\underline{x}}(y|x) \delta(P_{\underline{y}|\underline{x}}, 1) + (1 - P_{\underline{y}|\underline{x}}(y|x)) \delta(P_{\underline{y}|\underline{x}}, 0) \quad (43.108)$$

for all the leaf nodes $P_{\underline{y}|\underline{x}} \in \{0, 1\}$ of the image bipartite bnet. Also, let

⁵Note that we are using f_α to denote both a function $f_\alpha(\cdot)$ and a Boolean value. Which one we mean will be clear from context. f_α could also be used to denote, besides a function and a Boolean value, the real number $y_\alpha = f_\alpha(x_{nb(\underline{f}_\alpha)})$. However, we won’t be using it that third way in this chapter.

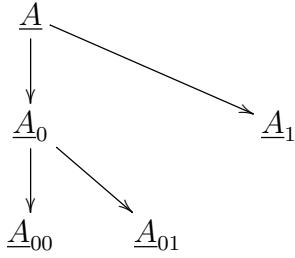


Figure 43.19: Example of a tree bnet.

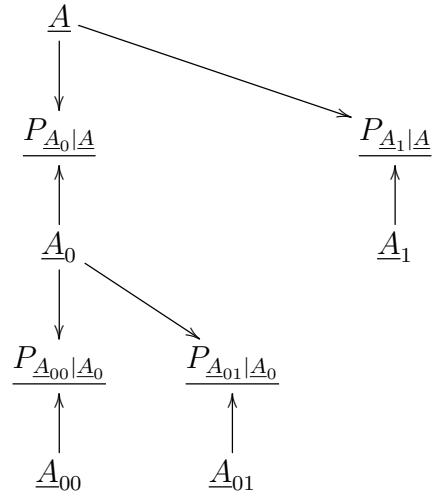


Figure 43.20: Bipartite bnet corresponding to tree bnet Fig.43.19.

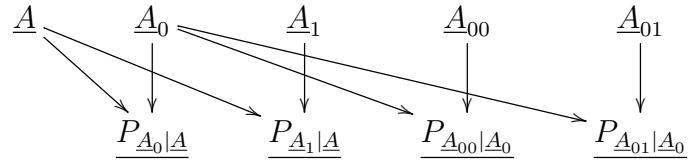


Figure 43.21: Different way of drawing the bnet Fig.43.20.

$$P_{\underline{y}}(y) = \text{arbitrary prior} \quad (43.109)$$

for all the root nodes \underline{y} of the image bipartite bnet except when \underline{y} corresponds to the root node \underline{A} of the source tree bnet. In that exceptional case,

$$P_{\underline{y}}(y) = P_{\underline{A}}(y) . \quad (43.110)$$

43.8 BP for bipartite bnets (BP-BB)

For a bipartite bnet as defined above, with root nodes \underline{x}_i and leaf nodes \underline{f}_α , let

$$nb(i) = \{\alpha : \underline{f}_\alpha \in nb(\underline{x}_i)\} , \quad (43.111)$$

$$nb(\alpha) = \{i : \underline{x}_i \in nb(\underline{f}_\alpha)\} , \quad (43.112)$$

$$m_{\alpha \leftarrow i}(x_i) = \pi_{\underline{f}_\alpha \leftarrow \underline{x}_i}(x_i) , \quad (43.113)$$

$$m_{\alpha \rightarrow i}(x_i) = \lambda_{\underline{f}_\alpha \rightarrow \underline{x}_i}(x_i) , \quad (43.114)$$

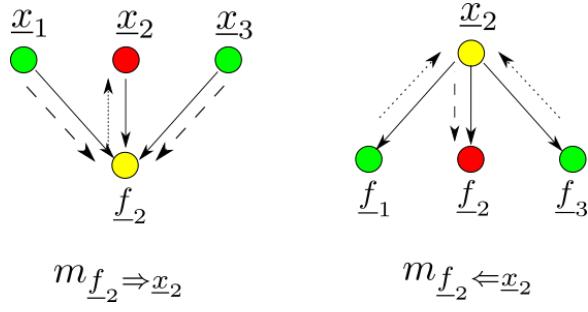


Figure 43.22: Fig.43.11 becomes this figure for the special case of a bipartite bnet. Union of green nodes and the red node = full neighborhood of yellow node. There are two possible cases: the red node is either a parent or a child of the yellow node.

Next we will show how to find $m_{\alpha \leftarrow i}^{(t)}$ and $m_{\alpha \rightarrow i}^{(t)}$ from $m_{\alpha \leftarrow i}^{(t-1)}$ and $m_{\alpha \rightarrow i}^{(t-1)}$.

1. Traversing an x (i.e., root) node.

See the $m_{\underline{f}_2 \leftarrow \underline{x}_2}$ panel of Fig.43.22.

For $i = 0, 1, \dots, nx - 1$, if $\alpha \in nb(i)$, then,

$$m_{\alpha \leftarrow i}^{(t)}(x_i) = \prod_{\beta \in nb(i) - \alpha} m_{\beta \rightarrow i}^{(t-1)}(x_i) , \quad (43.115)$$

whereas if $\alpha \notin nb(i)$

$$m_{\alpha \leftarrow i}^{(t)}(x_i) = m_{\alpha \leftarrow i}^{(t-1)}(x_i) . \quad (43.116)$$

2. Traversing an f (i.e., leaf) node.

See the $m_{\underline{f}_2 \Rightarrow \underline{x}_2}$ panel of Fig.43.22.

For $\alpha = 0, 1, \dots, nf - 1$, if $i \in nb(\alpha)$, then

$$m_{\alpha \Rightarrow i}^{(t)}(x_i) = \sum_{(x_k)_{k \in nb(\alpha)-i}} f_\alpha(x_{nb(\alpha)}) \prod_{k \in nb(\alpha)-i} m_{\alpha \Leftarrow k}^{(t-1)}(x_k) \quad (43.117)$$

$$= E_{(x_k)_{k \in nb(\alpha)-i}}^{(t-1)} [f_\alpha(x_{nb(\alpha)})] , \quad (43.118)$$

whereas if $i \notin nb(\alpha)$

$$m_{\alpha \Rightarrow i}^{(t)}(x_i) = m_{\alpha \Rightarrow i}^{(t-1)}(x_i) . \quad (43.119)$$

In the above equations, if the range set of a product is empty, then define the product as 1; i.e., $\prod_{k \in \emptyset} F(k) = 1$.

Claim:

$$P(x_i | \epsilon) = \lim_{t \rightarrow \infty} \mathcal{N}(!x_i) \prod_{\alpha \in nb(i)} m_{\alpha \Rightarrow i}^{(t)}(x_i) \quad (43.120)$$

and

$$P(x_{nb(\alpha)} | \epsilon) = \lim_{t \rightarrow \infty} \mathcal{N}(!x_{nb(\alpha)}) f_\alpha(x_{nb(\alpha)}) \prod_{k \in nb(\alpha)} m_{\alpha \Leftarrow k}^{(t)}(x_k) . \quad (43.121)$$

43.8.1 BP-BB and general BP agree on Markov chains

It is instructive to compare the belief values (i.e., $P(x_i | \epsilon)$) obtained by applying the general (i.e., polytree) BP and BP-BB algorithms to a Markov chain. Next we show that both algorithms yield the same belief values.

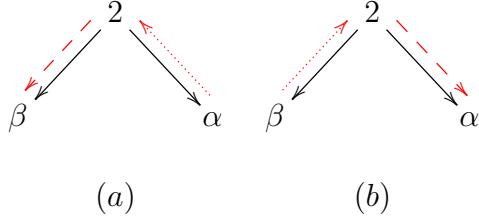


Figure 43.23: Traversing a root node of a Markov chain (a)Propagation towards left (i.e., towards future). (b)Propagation towards right (i.e., towards past).

Consider the BP-BB rule for traversing a root node. When traveling towards the left as in Fig.43.23 (a), it implies that

$$m_{\alpha \Rightarrow 2}(x_2) = m_{\beta \Leftarrow 2}(x_2) , \quad (43.122)$$

and when traveling towards the right as in Fig.43.23 (b), it implies that

$$m_{\beta \Rightarrow 2}(x_2) = m_{\alpha \Leftarrow 2}(x_2) . \quad (43.123)$$

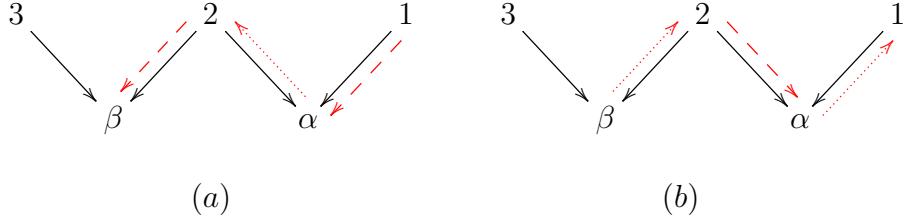


Figure 43.24: Traversing a leaf node of a Markov chain (a)Propagation towards left (i.e., towards future). (b)Propagation towards right (i.e., towards past).

Now consider the BP-BB rule for traversing a leaf node. When traveling to the left as in Fig.43.24 (a), it implies that

$$\underbrace{m_{\alpha \Rightarrow 2}(x_2)}_{\lambda} = \sum_{x_1} P(x_2|x_1) \underbrace{m_{\alpha \Leftarrow 1}(x_1)}_{\pi} . \quad (43.124)$$

One can rewrite the left and right hand sides (LHS, RHS) of Eq.(43.124) as follows

$$RHS = \sum_{x_1} P(x_2|x_1) \pi_{\alpha \Leftarrow 1}(x_1) , \quad (43.125)$$

and

$$LHS = m_{\alpha \Rightarrow 2}(x_2) = m_{\beta \Leftarrow 2}(x_2) = \pi_{\beta \Leftarrow 2}(x_2) , \quad (43.126)$$

Therefore

$$\pi_{\beta \Leftarrow 2}(x_2) \sum_{x_1} P(x_2|x_1) \pi_{\alpha \Leftarrow 1}(x_1) . \quad (43.127)$$

Once again, consider the BP-BB rule for traversing a leaf node. When traveling to the right as in Fig.43.24 (b), it implies that

$$\underbrace{m_{\alpha \Rightarrow 1}(x_1)}_{\lambda} = \sum_{x_2} P(x_2|x_1) \underbrace{m_{\alpha \Leftarrow 2}(x_2)}_{\pi} . \quad (43.128)$$

One can rewrite the left and right hand sides (LHS, RHS) of Eq.(43.128) as follows

$$RHS = \sum_{x_2} P(x_2|x_1) \pi_{\alpha \Leftarrow 2}(x_2) \quad (43.129)$$

$$= \sum_{x_2} P(x_2|x_1) \lambda_{\beta \Rightarrow 2}(x_2) , \quad (43.130)$$

and

$$LHS = \lambda_{\alpha \Rightarrow 1}(x_1) . \quad (43.131)$$

Therefore,

$$\lambda_{\alpha \Rightarrow 1}(x_1) = \sum_{x_2} P(x_2|x_1) \lambda_{\beta \Rightarrow 2}(x_2) . \quad (43.132)$$

Finally, note that Eq.(43.120) becomes

$$P(x_2|\epsilon) = \mathcal{N}(!x_2) m_{\beta \Rightarrow 2}(x_2) m_{\alpha \Rightarrow 2}(x_2) \quad (43.133)$$

$$= \mathcal{N}(!x_2) m_{\alpha \Leftarrow 2}(x_2) m_{\alpha \Rightarrow 2}(x_2) \quad (43.134)$$

$$= \mathcal{N}(!x_2) \pi_{\alpha \Leftarrow 2}(x_2) \lambda_{\alpha \Rightarrow 2}(x_2) \quad (43.135)$$

$$= \mathcal{N}(!x_2) P(x_2|\epsilon^-) P(x_2|\epsilon^+) \quad (43.136)$$

and Eq.(43.121) becomes

$$P(x_2, x_1) = \mathcal{N}(!x_2, !x_1) P(x_2|x_1) m_{\alpha \Leftarrow 1}(x_1) m_{\alpha \Leftarrow 2}(x_2) \quad (43.137)$$

$$= \mathcal{N}(!x_2, !x_1) P(x_2|x_1) \pi_{\alpha \Leftarrow 1}(x_1) \pi_{\alpha \Leftarrow 2}(x_2) . \quad (43.138)$$

43.8.2 BP-BB and general BP agree on tree bnets.

It is instructive to compare the belief values (i.e., $P(x_i|\epsilon)$) obtained by applying the general (i.e., polytree) BP and BP-BB algorithms to a tree bnet. Next we show that both algorithms yield the same belief values.

Applying to the left panel of Fig.43.25 the BP-BB rule for traversing a root node, we get

$$m_{\alpha \Leftarrow \underline{x}}(x) = \prod_i m_{\beta_i \Rightarrow \underline{x}}(x) . \quad (43.139)$$

Applying to the left panel of Fig.43.25 the BP-BB rule for traversing a leaf node, we get

$$m_{\alpha \Rightarrow \underline{a}}(a) = \mathcal{N}(!a) \sum_x m_{\alpha \Leftarrow \underline{x}}(x) P(x|a) . \quad (43.140)$$

Combining Eqs.(43.139) and (43.140), we get

$$m_{\alpha \Rightarrow \underline{a}}(a) = \mathcal{N}(!a) \sum_x P(x|a) \prod_i m_{\beta_i \Rightarrow \underline{x}}(x) , \quad (43.141)$$

which can be rewritten as

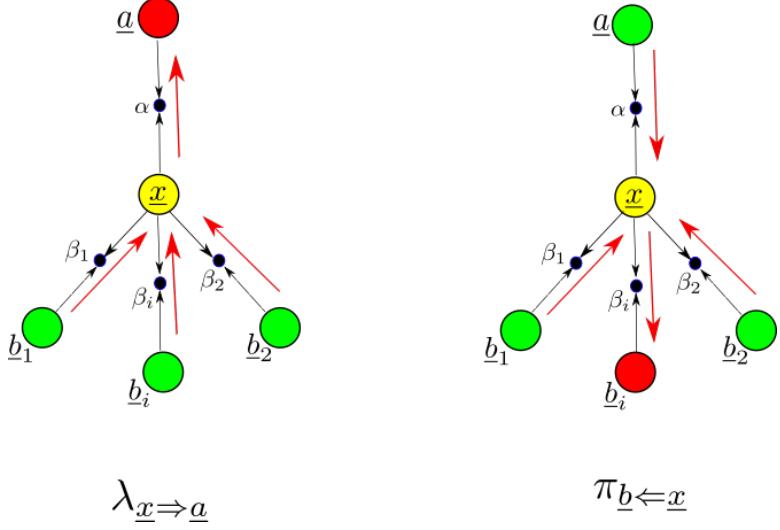


Figure 43.25: Subgraph of a tree bnet. This is the same as Fig.43.11, except that here the yellow node has a single parent because this is a subgraph of a tree bnet, not of an arbitrary bnet like Fig.43.11. The subgraph has been converted to a subgraph of a bipartite bnet by inserting a collider leaf node, labeled by a Greek letter, at the center of each edge of the tree bnet. Red arrows indicate the direction of message info flow.

$$\lambda_{\underline{x} \Rightarrow \underline{a}}(a) = \mathcal{N}(!a) \sum_x P(x|a) \underbrace{\prod_i \lambda_{\underline{b}_i \Rightarrow \underline{x}}(x)}_{\lambda_{\underline{x}}(x)} . \quad (43.142)$$

Eq.(43.142) is just RULE 1 for general BP.

Applying to the right panel of Fig.43.25 the BP-BB rule for traversing a root node, we get

$$m_{\beta_i \Leftarrow \underline{x}}(x) = \mathcal{N}(!x)m_{\alpha \Rightarrow \underline{x}}(x) \prod_{k \neq i} m_{\beta_k \Rightarrow \underline{x}}(x) \quad (43.143)$$

Applying to the right panel of Fig.43.25 the BP-BB rule for traversing a leaf node, we get

$$m_{\alpha \Rightarrow \underline{x}}(x) = \sum_a P(x|a)m_{\alpha \Leftarrow a}(a) \quad (43.144)$$

$$= \sum_a P(x|a)\pi_{\underline{x} \Leftarrow a}(a) \quad (43.145)$$

$$= \pi_{\underline{x}}(x) . \quad (43.146)$$

Combining Eqs.(43.143) and (43.146), we get

$$\pi_{\underline{b}_i \Leftarrow \underline{x}}(x) = \mathcal{N}(!x)\pi_{\underline{x}}(x) \prod_{k \neq i} \lambda_{b_k \Rightarrow \underline{x}}(x) . \quad (43.147)$$

Eq.(43.147) is just RULE 2 of general BP.

43.9 BP-BB and sum-product decomposition

BP-BB yields what is often referred to as a **sum-product decomposition**. I don't like that name because it is unnecessarily confusing, and it fails to convey the recursive nature⁶ of the decomposition. I prefer to call it a **recursive sum of products (RSOP) decomposition**, and will call it so henceforth in this chapter.

Expressing the marginals of a bnet as RSOPs, which is what BP does, reduces the complexity of the calculation. (i.e., the total number of additions and multiplications that need to be performed) That makes using the BP algo very advantageous. For instance, consider a Markov chain $\underline{x}_{n-1} \leftarrow \dots \leftarrow \underline{x}_1 \leftarrow \underline{x}_0$, where $x_i \in \{0, 1, 2\}$ for all i . Note that if we calculate $P(x_{n-1})$ as follows

$$P(x_{n-1}) = \left[\sum_{x_{n-2}} P(x_{n-1}|x_{n-2}) \dots \left[\sum_{x_1} P(x_2|x_1) \left[\sum_{x_0} P(x_1|x_0)P(x_0) \right] \dots \right] \right] , \quad (43.148)$$

we need to perform $2(n - 1)$ additions and $3(n - 1)$ multiplications. On the other hand, if we calculate $P(x_{n-1})$ as follows

$$P(x_{n-1}) = \sum_{x_{n-2}} \dots \sum_{x_1} \sum_{x_0} P(x_{n-1}|x_{n-2}) \dots P(x_2|x_1)P(x_1|x_0)P(x_0) , \quad (43.149)$$

we need to perform $3^n - 1$ additions and $3^n(n - 1)$ multiplications.

⁶By "recursive nature", we mean bootstrapped definitions that lead to nested sums. The recursive nature of BP is evident from RULES 1 and 2 that define λ 's and π 's in terms of other λ 's and π 's.

Chapter 44

Message Passing (Belief Propagation) in Quantum Mechanics

See Ref.[61].

Chapter 45

Meta-learners for estimating ATE

This section is based on the final 2 chapters of Ref.[8].

The Average Treatment Effect (ATE) is defined in Chapter 56.

Economists are huge fans of Linear Regression (LR), and traditionally calculate ATE using LR. But in recent times, they have begun to calculate ATE using Machine Learning (ML) instead. This chapter describes various methods that economists and others have devised for calculating ATE with ML. These methods are called **meta-learners** because they involve multiple ML or LR steps.

Using ML to calculate ATE captures non-linear trends whose exclusion might sometimes lead to a poor result. On the other hand, ML is more expensive computationally than LR, and it introduces the danger of overfitting, a danger which is nonexistent with LR.

Below, we represent each Linear Regression (LR) step as follows. We list a dataset; i.e., a set of tuples indexed by the individuals σ of a population Σ such that $|\Sigma| = nsam$. The independent variables of the LR (i.e., x^σ) are unboxed and the dependent variable (aka target feature) (i.e., y^σ) is shown inside a box. Then we show an arrow with the superscript “LR-fit”, followed by the fit function obtained by performing the LR.¹

$$\{(\sigma, x^\sigma = [x_i^\sigma], \boxed{y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{LR-fit}} \hat{y}(x) = \alpha + x_i \beta_i \quad (45.1)$$

Analogously, below, we represent each Supervised Machine Learning (ML) step as follows.

$$\{(\sigma, x^\sigma, \boxed{y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \hat{y}(x) \quad (45.2)$$

Henceforth, we will use $\delta(x) (\approx \text{ATE})$ to denote the treatment effect.

- **S (Single)-learner**

¹In this chapter, we will occasionally use the Einstein summation convention; i.e., implicit sum over repeated indices.

$$\{(\sigma, x^\sigma, d^\sigma, \boxed{y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \hat{y}(d, x) \quad (45.3)$$

$$\delta(x) = \hat{y}(1, x) - \hat{y}(0, x) \quad (45.4)$$

- **T (Twin)-learner**

$$\{(\sigma, x^\sigma, d^\sigma = 0, \boxed{y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \hat{y}_0(x) \quad (45.5)$$

$$\{(\sigma, x^\sigma, d^\sigma = 1, \boxed{y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \hat{y}_1(x) \quad (45.6)$$

$$\delta(x) = \hat{y}_1(x) - \hat{y}_0(x) \quad (45.7)$$

- **X (Cross)-learner**

do T-learner, get $\hat{y}_0(x), \hat{y}_1(x)$

$$\{(\sigma, x^\sigma, d^\sigma = 0, \boxed{y^\sigma - \hat{y}_1(x^\sigma)}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \mathcal{Y}_0(x) \quad (45.8)$$

$$\{(\sigma, x^\sigma, d^\sigma = 1, \boxed{y^\sigma - \hat{y}_0(x^\sigma)}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \mathcal{Y}_1(x) \quad (45.9)$$

$$\delta(x) = \frac{1}{2}[\mathcal{Y}_1(x) - \mathcal{Y}_0(x)] \quad (45.10)$$

- **De-biased (aka Orthogonal) ML**

Standard supervised ML is performed with two features, the independent feature and the dependent or target feature. Supervised ML has a target feature. Unsupervised ML doesn't.

In **De-biased LR**, we do LR with two residuals. Let's call them the **independent residual** and the **dependent or target residual**. These two residuals are calculated with the help of two previously performed LR steps.

In **De-biased ML**, we do ML or LR (either one) with two residuals. These two residuals are calculated with the help of two previously performed ML steps (instead of two LR steps).

The FWL theorem discussed in Chapter 24 shows how to do De-biased LR. Next, we will describe how to do De-biased ML.

We start by doing two ML steps:

$$\{(\sigma, x^\sigma, \boxed{d^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \hat{d}(x) \quad (45.11)$$

$$\{(\sigma, x^\sigma, \boxed{y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \hat{y}(x) \quad (45.12)$$

After these two ML steps, we do either LR or ML to get $\delta(x)$.

Let

$$\Delta d^\sigma = d^\sigma - \hat{d}(x^\sigma) \quad (45.13)$$

$$\Delta y^\sigma = y^\sigma - \hat{y}(x^\sigma) \quad (45.14)$$

Options for the final learning step that calculates $\delta(x)$:

1. Do a standard LR to get a $\delta(x)$ that is a constant (i.e., x independent):

$$\{(\sigma, \Delta d^\sigma, \boxed{\Delta y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{LR-fit}} \mathcal{Y}(\Delta d) = \alpha + \Delta d \delta \quad (45.15)$$

$$\delta = \text{coefficient of } \Delta d \text{ in } \mathcal{Y}(\Delta d). \quad (45.16)$$

2. Do a LR with a $x * d$ cross term to get a $\delta(x)$ that is linear in x :

$$\{(\sigma, x^\sigma, \Delta d^\sigma, \boxed{\Delta y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{LR-fit}} \mathcal{Y}(\Delta d, x) = \alpha_0 + x\beta_0 + \Delta d(\alpha_1 + x\beta_1) \quad (45.17)$$

$$\delta(x) = \mathcal{Y}(\Delta d|_{d=1}, x) - \mathcal{Y}(\Delta d|_{d=0}, x) = \alpha_1 + x\beta_1 \quad (45.18)$$

3. Do ML with a weighted cost function to get a general fit $\hat{\delta}(x)$.

The cost function used in standard ML (see Eq.(45.2)) is:

$$\mathcal{C} = \frac{1}{nsam} \sum_{\sigma} [y^\sigma - \hat{y}(x^\sigma)]^2 \quad (45.19)$$

Define the cost function in this case as

$$\mathcal{C} = \frac{1}{nsam} \sum_{\sigma} \left[\Delta y^\sigma - \hat{\delta}(x^\sigma) \Delta d^\sigma \right]^2 \quad (45.20)$$

$$= \frac{1}{nsam} \sum_{\sigma} [\Delta d^\sigma]^2 \left[\frac{\Delta y^\sigma}{\Delta d^\sigma} - \hat{\delta}(x^\sigma) \right]^2 \quad (45.21)$$

This is a weighted cost function with weights $[\Delta d^\sigma]^2$.

$$\{(\sigma, x^\sigma, \boxed{\frac{\Delta y^\sigma}{\Delta d^\sigma}}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \hat{\delta}(x) \quad (45.22)$$

4. Do ML with Δd^σ as an independent feature to get a general fit $\hat{\delta}(\Delta d, x)$:

$$\{(\sigma, x^\sigma, \Delta d^\sigma, \boxed{\frac{\Delta y^\sigma}{\Delta d^\sigma}}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \hat{\delta}(\Delta d, x) \quad (45.23)$$

$$\delta(x) = \hat{\delta}(\Delta d|_{d=1}, x) \quad (45.24)$$

Chapter 46

Missing Data, Imputation

This chapter assumes that the reader has read some parts of Chapter 23 on the Expectation Maximization (EM) algo and Chapter 40 on Markov Chain Monte Carlo (MCMC).

	h_0	x_0	x_1	x_2	
1	NA	0	1	1	
2	NA	0	0	0	
3	NA	1	1	0	
4	NA	NA	1	NA	
5	NA	0	NA	1	
6	NA	0	0	1	

	h_0	x_0	x_1	x_2	m
1	NA	0	1	1	(0,0,0)
2	NA	0	0	0	(0,0,0)
3	NA	1	1	0	(0,0,0)
4	NA	0		0	
		0		1	
		1		0	(1,0,1)
		1		1	
5	NA	0	0	1	(0,1,0)
6	NA	0	0	1	(0,0,0)

Table 46.1: **Left Table:** Dataset with $nsam = 6$ and some missing entries, for 4 binary variables h_0, x_0, x_1, x_2 . NA=not available. The h_0 column is completely missing because h_0 is an unobserved variable. **Right Table:** All possibilities for $x_i = NA$ cells of left table have been enumerated. A new column labeled m has been added. $m_i = \mathbb{1}(x_i \text{ is missing})$ for $i = 0, 1, 2$.

Suppose that you have compiled a **dataset** $\vec{x} = (x[\sigma])_{\sigma=0,1,\dots,nsam-1}$ where $x = (x_0, x_1, \dots, x_{nx-1})$ from a study or survey. It consists of $nsam$ number of samples (sample= row), and nx columns (each column is a different feature, or observation). Suppose that some of the cells in this matrix are empty. Throwing away all the incomplete rows is okay if the number of incomplete rows is much smaller than $nsam$. If not, throwing them away would throw away a substantial amount of information contained in all the filled cells in those incomplete rows, plus it might bias your dataset. This chapter deals with how to fill those empty cells with plausible fake data. A fancy name for this process is **imputation**. There is no unique way of

fabricating fake data, but some fakes are better than others by some metrics. This chapter will consider two popular ways (EM and MCMC) of filling those empty cells with their “most likely” values based on the cells of the dataset that aren’t missing, and also based on some bnet model that is expected to describe well the dataset.

Notation: $\vec{a} = (\underline{a}[\sigma])_{\sigma=0,1,\dots,nsam-1}$, where $nsam$ is the number of samples. Will sometimes denote $a[\sigma]$ by a^σ .

For concreteness, we will apply the concepts of this chapter to the dataset with missing data given by Table 46.1.

46.1 Imputation via EM

We begin by augmenting Fig.23.1 (the first figure in Chapter 23) by adding to it a new node \vec{m} called the **missingness variable**. Recall that node θ represents the **unknown parameters**, node \vec{x} represents the **observed variables**, and node \vec{h} represents the **hidden variables**. Both θ and \vec{h} are hidden (i.e., unobserved). Fig.46.1 shows 3 popular ways of connecting node \vec{m} to the other nodes in the graph Fig.23.1.

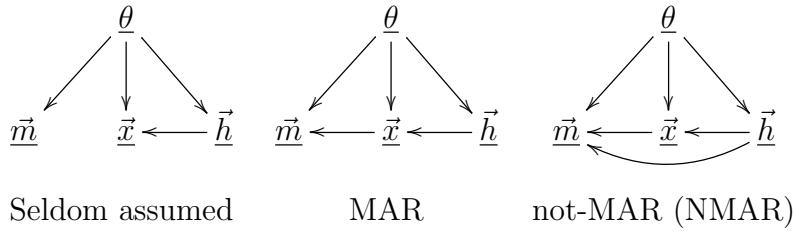


Figure 46.1: The left bnet is seldom assumed. The middle bnet is referred to as the MAR (missing at random) assumption. The right bnet is referred to as the not-MAR (NMAR) assumption.

From Fig.46.1, we have

$$P(\vec{m}|\vec{x}, \vec{h}, \theta) = \begin{cases} P(\vec{m}|\theta) & \text{Seldom assumed. Called missing-CAR (MCAR)} \\ P(\vec{m}|\vec{x}, \theta) & \text{MAR} \\ P(\vec{m}|\vec{x}, \vec{h}, \theta) & \text{not-MAR (NMAR)} \end{cases}. \quad (46.1)$$

For doing imputation via EM, we connect node \vec{m} as shown in the middle bnet (called MAR) of Fig.46.1.

For the example of Table 46.1, we have variables \vec{m} , \vec{x} and \vec{h} whose values range over the following sets:

$$\begin{aligned} \vec{x} &= (\vec{x}_0, \vec{x}_1, \vec{x}_2) \\ \vec{h} &= (\vec{h}_0) \\ \underline{h}_0[\sigma] &\in \{0, 1\}, \end{aligned}$$

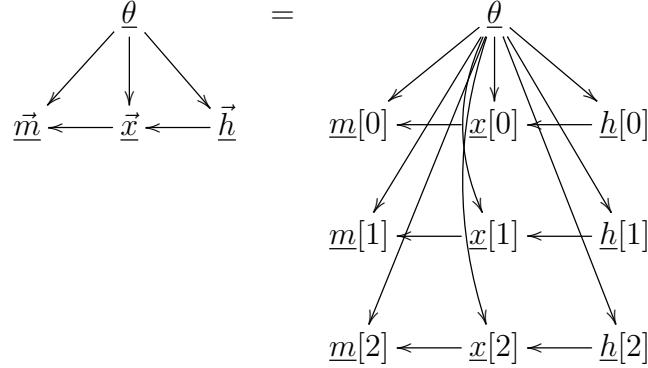


Figure 46.2: MAR bnet with $nsam = 3$.

$$\begin{aligned} \underline{x}_i[\sigma] &\in \{0, 1\} \text{ for } i = 0, 1, 2, \\ \underline{m}_i[\sigma] &\in \{0, 1\} \text{ for } i = 0, 1, 2. \end{aligned}$$

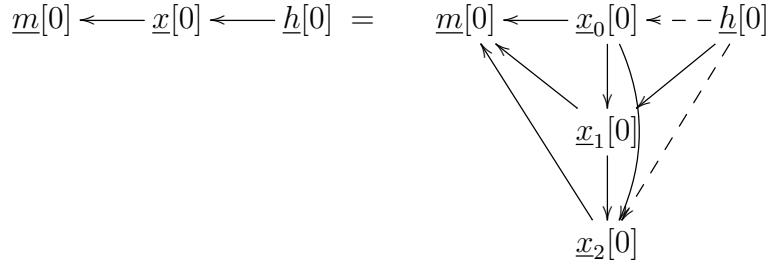


Figure 46.3: Our example for imputation via EM assumes this bnet between nodes $m[\sigma], x[\sigma], h[\sigma]$.

For concreteness, we will assume that the Markov chain $\underline{m}[\sigma] \leftarrow \underline{x}[\sigma] \leftarrow \underline{h}[\sigma]$ has a finer grained DAG structure given by Fig.46.3. where we will omit the dashed arrows. If one doesn't want to assume that the data can be fitted well by the bnet of Fig.46.3 without the dashed arrows, one can include those arrows too, at the expense of more unknown parameters (i.e., degrees of freedom) to be lumped into θ . We will parameterize the TPMs corresponding to Fig.46.3 using a Categorical Distribution for each column of the TPMs. We will thus assume that the TPMs, printed in blue, for bnet Fig.46.3, are as follows.

$$P(h_0^\sigma | \theta) = \frac{1 - \theta_0}{1 + \theta_0} \quad (46.2)$$

$$P(x_0^\sigma | \theta) = \begin{array}{c|cc} & & \\ & 0 & 1 - \theta_1 \\ \hline 0 & & \\ & 1 & \theta_1 \end{array} \quad (46.3)$$

$$P(x_1^\sigma | x_0^\sigma, h^\sigma, \theta) = \begin{array}{c|cccc} & 00 & 01 & 10 & 11 \\ \hline 0 & 1 - \theta_2 & 1 - \theta_3 & 1 - \theta_4 & 1 - \theta_5 \\ 1 & \theta_2 & \theta_3 & \theta_4 & \theta_5 \end{array} \quad (46.4)$$

$$P(x_2^\sigma | x_1^\sigma, x_0^\sigma, h^\sigma, \theta) = \begin{array}{c|cccc} & 00 & 01 & 10 & 11 \\ \hline 0 & 1 - \theta_6 & 1 - \theta_7 & 1 - \theta_8 & 1 - \theta_9 \\ 1 & \theta_6 & \theta_7 & \theta_8 & \theta_9 \end{array} \quad (46.5)$$

$$P(m^\sigma | x^\sigma, \theta) = \frac{1}{nsam} P((x_i)_{\forall i \ni m_i=1} | (x_i)_{\forall i \ni m_i=0}, \theta) \quad (46.6)$$

Eq.(46.6) can be illustrated as follows. In Table 46.2, we added a $P(m)$ column to Table 46.1.

	h_0	x_0	x_1	x_2	m	$P(m)$
1	NA	0	1	1	(0,0,0)	$\frac{1}{nsam}$
2	NA	0	0	0	(0,0,0)	$\frac{1}{nsam}$
3	NA	1	1	0	(0,0,0)	$\frac{1}{nsam}$
4	NA	0		0		$\frac{1}{nsam} P(x_0 = 0, x_2 = 0 x_1 = 1, \theta)$
		0		1		$\frac{1}{nsam} P(x_0 = 0, x_2 = 1 x_1 = 1, \theta)$
		1		0		$\frac{1}{nsam} P(x_0 = 1, x_2 = 0 x_1 = 1, \theta)$
		1		1		$\frac{1}{nsam} P(x_0 = 1, x_2 = 1 x_1 = 1, \theta)$
5	NA	0	0		(0,1,0)	$\frac{1}{nsam} P(x_1 = 0 x_0 = 0, x_2 = 1, \theta)$
		1		1		$\frac{1}{nsam} P(x_1 = 1 x_0 = 0, x_2 = 1, \theta)$
6	NA	0	0	1	(0,0,0)	$\frac{1}{nsam}$

Table 46.2: $P(m)$ column added to Table 46.1. Note that $\sum_m P(m) = 1$.

$$\theta = (\theta_i)_{i=0,1,\dots,9} \quad (46.7)$$

$$P(m^\sigma, x^\sigma, h^\sigma | \theta) = P(m^\sigma | x^\sigma, \theta) P(x^\sigma | h^\sigma, \theta) P(h^\sigma | \theta) \quad (46.8)$$

$$P(x^\sigma | h^\sigma, \theta) = P(x_2^\sigma | x_1^\sigma, x_0^\sigma, \theta) P(x_1^\sigma | x_0^\sigma, h^\sigma, \theta) P(x_0^\sigma | \theta) \quad (46.9)$$

$$P(x_1^\sigma | x_0^\sigma, \theta) = \sum_h P(x_1^\sigma | x_0^\sigma, h^\sigma, \theta) P(h^\sigma | \theta) \quad (46.10)$$

$$P(x^\sigma|\theta) = P(x_2^\sigma|x_1^\sigma, x_0^\sigma, \theta)P(x_1^\sigma|x_0^\sigma, \theta)P(x_0^\sigma|\theta) \quad (46.11)$$

$$Q(\theta|\theta^{(t)}) = \sum_{\vec{m}, \vec{h}} P(\vec{m}, \vec{h} | \vec{x}, \theta^{(t)}) \ln P(\vec{m}, \vec{x}, \vec{h}|\theta) \quad (46.12)$$

$$= \sum_{\vec{m}, \vec{h}} \left[\prod_{\sigma} P(m^\sigma, h^\sigma | x^\sigma, \theta^{(t)}) \right] \ln \left[\prod_{\sigma} P(m^\sigma, x^\sigma, h^\sigma|\theta) \right] \quad (46.13)$$

$$= \sum_{\sigma} \sum_{m^\sigma, h^\sigma} P(m^\sigma, h^\sigma | x^\sigma, \theta^{(t)}) \ln P(m^\sigma, x^\sigma, h^\sigma|\theta) \quad (46.14)$$

$$= \sum_{\sigma} \sum_{m^\sigma, h^\sigma} \frac{P(m^\sigma, h^\sigma, x^\sigma | \theta^{(t)})}{P(x^\sigma | \theta^{(t)})} \ln P(m^\sigma, x^\sigma, h^\sigma|\theta) \quad (46.15)$$

Once you find optimal parameters θ^* by recursing this $Q(\theta|\theta^{(t)})$, you can evaluate numerically the $P(m)$ column of Table 46.2. In Table 46.2, out of the 4 sub-rows for row 4, choose the one with the highest probability. Similarly, out of the 2 sub-rows for row 5, choose the one with the highest probability.

46.2 Imputation via MCMC

A simple and popular way to do imputation via MCMC is described in Ref.[54]. It goes as follows.

Let

$$\underline{H}[\sigma] = (\underline{h}[\sigma], \underline{m}[\sigma]) \quad (46.16)$$

for $\sigma = 0, 1, \dots, nsam - 1$. Initialize $\theta^{(0)}$ to a random value within the allowed ranges. Do the following 2 steps, for $t = 0, 1, \dots, T - 1$, where T is large enough that $\theta^{(t)}$ has reached a steady value that is independent of $\theta^{(0)}$. To do the sampling, use a standard sampling technique such as Gibbs sampling.

- **STEP 1:** For $\sigma = 0, 1, \dots, nsam - 1$, find a sample

$$(H^\sigma)^{(t+1)} \sim P(H^\sigma | x^\sigma, \theta^{(t)}) . \quad (46.17a)$$

- **STEP 2:** Find a sample

$$\theta^{(t+1)} \sim P^{(t+1)}(\theta) \quad (46.17b)$$

where

$$P^{(t+1)}(\theta) = \mathcal{N}(!\theta)P(\vec{x}, \vec{H}^{(t+1)}|\theta) \quad (46.17c)$$

$$= \mathcal{N}(!\theta) \prod_{\sigma} P(x^\sigma, (H^\sigma)^{(t+1)}|\theta) . \quad (46.17d)$$

Fig.46.4 illustrates this two step recursive process using a bnet.

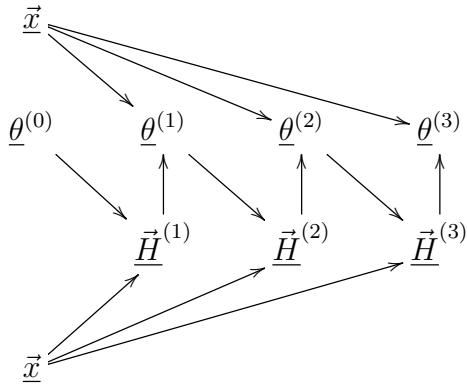


Figure 46.4: bnet illustrating Eqs.(46.17) for doing imputation via MCMC. The **same** node \underline{x} appears twice to make the graph clearer.

46.3 Multiple Imputations

Multiple imputations means calculating θ^* (i.e., the optimum θ) and the concomitant dataset \vec{x}^*, \vec{H}^* , via any method (such as EM or MCMC), a large number of times, starting from different, randomly chosen $\theta^{(0)}$ initial parameters. Then calculating the average and the variance of $\theta^*, \vec{x}^*, \vec{H}^*$ and functions thereof.

Chapter 47

Monty Hall Problem

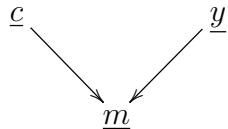


Figure 47.1: Monty Hall Problem.

Mr. Monty Hall, host of the game show “Let’s Make a Deal”, hides a car behind one of three doors and a goat behind each of the other two. The contestant picks Door No. 1, but before opening it, Mr. Hall opens Door No. 2 to reveal a goat. Should the contestant stick with No. 1 or switch to No. 3?

The Monty Hall problem can be modeled by the bnet Fig.47.1, where

- \underline{c} = the door behind which the car actually is.
- \underline{y} = the door opened by you (the contestant), on your first selection.
- \underline{m} = the door opened by Monty (game host)

We label the doors 1,2,3 so $S_c = S_y = S_m = \{1, 2, 3\}$.

The TPMs, printed in blue, for this bnet, are as follows:

$$P(c) = \frac{1}{3} \text{ for all } c \quad (47.1)$$

$$P(y) = \frac{1}{3} \text{ for all } y \quad (47.2)$$

$$P(m|c, y) = \mathbb{1}(m \neq c) \left[\frac{1}{2} \mathbb{1}(y = c) + \mathbb{1}(y \neq c) \mathbb{1}(m \neq y) \right] \quad (47.3)$$

It's easy to show that the above node probabilities imply that

$$P(c = 1|m = 2, y = 1) = \frac{1}{3} \quad (47.4)$$

$$P(c = 3|m = 2, y = 1) = \frac{2}{3} \quad (47.5)$$

So you are twice as likely to win if you switch your final selection to be the door which is neither your first choice nor Monty's choice.

The way I justify this to myself is: Monty gives you a piece of information. If you don't switch your choice, you are wasting that info, whereas if you switch, you are using the info.

Chapter 48

Multi-armed Bandits

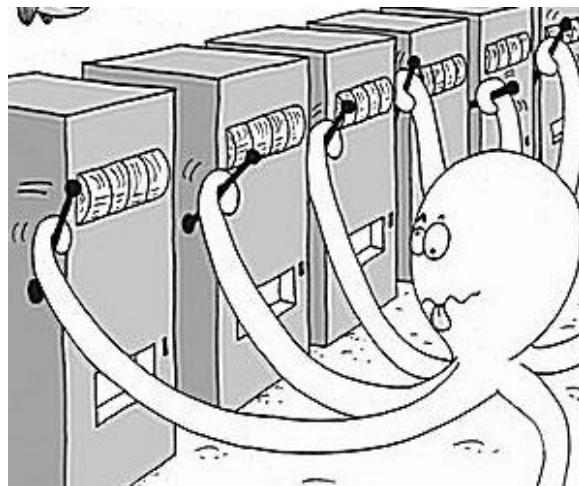


Figure 48.1: Multi-armed bandit (MAB).

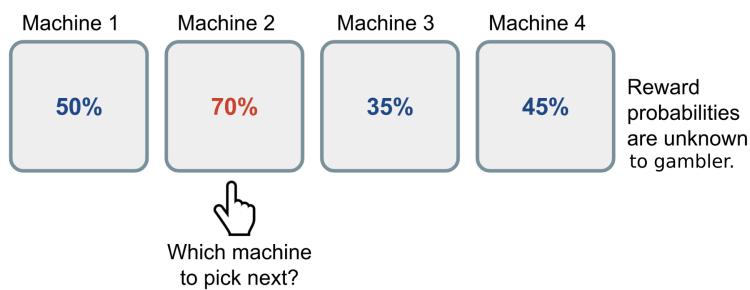


Figure 48.2: Bernoulli MAB (Bern-MAB) with 4 arms. For a Bern-MAB, the conditional probability distribution $P(r|a)$ for reward r , given arm a , is initially totally unknown to the gambler (agent), but it is known by the environment to be a Bernoulli distribution $P(r|a) = \mu_a^r(1 - \mu_a)^{1-r}$ for $r \in \{0, 1\}$, where $0 < \mu_a < 1$ for each a . The percentages shown are μ_a for each arm a .

This chapter is mostly based on Refs. [45] and [64].

Multi-armed Bandits (MABs) are a simple version of Reinforcement Learning (RL). RL is discussed in Chapter 61.

The term “one-armed-bandit” is a humorous term for what is also called a slot machine. A slot machine is a gambling device which has a slot into which you put coins or tokens for the privilege of being allowed to pull down a lever (arm) on one side of the device. This action generates a random combination of three shapes, which may or may not, depending on their combination, entitle the player to a money award.

Multi-armed bandit (MAB) is the name given to the optimization problem that considers an agent (gambler) that is playing multiple one-armed-bandits, each with a possibly different odds of winning. The optimization problem is to determine an efficient schedule whereby the gambler can converge on the device with the highest odds of winning.

MABs are often used in marketing as an alternative to A/B testing. These 2 methods yield different information but overlap in that they both can discover consumer preferences.

The MAB problem is an optimization problem (i.e., finding the maximum of a reward function or the minimum of a cost function). As with any minimization problem, an algorithm to solve it runs the danger of converging to a local minimum that isn’t the global (i.e., the overall) minimum. This danger can be diminished by doing both **exploration and exploitation**. Algorithms that do no exploration, only exploitation, are said to be **greedy**, and they are at the highest risk of converging to a non-global minimum.

48.1 Bnet for MAB

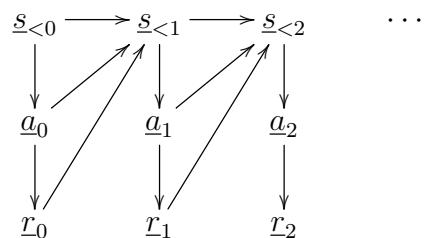


Figure 48.3: Bnet for a multi-armed bandit (MAB).

Let

$t \in \{0, 1, 2, \dots\}$ be the **time** slice (step),

$\underline{a}_t \in \{0, 1, \dots, N_a - 1\} = S_a$ be the **arm** of bandit that is pulled at time t , out of N_a arms. In RL language, it's also the **action** taken.

$\underline{r}_t \in \mathbb{R}$ be the **reward** at time t ,

$s_{<t} = \underbrace{[(a_\tau, r_\tau) : \tau < t]}_{s_\tau}$ be the **state** at time t .

Fig.48.3 shows a bnet for a MAB. The TPMs, printed in blue, for this bnet, are as follows.

$$P(s_{<t}|s_{<t-1}, a_{t-1}, r_{t-1}) = \mathbb{1}(s_{<t} = s_{<t-1} \cup (a_{t-1}, r_{t-1})) \quad (48.1)$$

For $t = 0$, $s_{<0} = \emptyset$, so choose a random a_0 .

$$P(a_t|s_{<t}) = \mathbb{1}(a_t = a_t^*) \text{ (agent's response)} , \quad (48.2)$$

where a_t^* depends on the **strategy (aka policy)** being used by the **gambler (agent)**. We consider various strategies below. a_t^* is defined solely in terms of empirical data (i.e., a_t, r_t values) collected from previous time slices. That is the only type of information that the gambler is privy to. a_0^* is chosen at random. The formulae for a_t^* that we give below for various strategies should only be used for $t > 0$.

$$P(r_t|a_t) = P_{r|a}(r_t|a_t) \text{ (environment's response)} . \quad (48.3)$$

$P_{r|a}$ is a probabilistic model that models the **environment** in which the agent lives. This assumes that the a_t (and the r_t) are i.i.d. $P_{r|a}$ depends on parameters whose values are known by the environment, but are not known a priori by the gambler. In fact, the goal of this exercise is for the gambler to find ever more accurate estimates of those parameters, using only the empirical data he/she can collect from the past, starting from total ignorance about those parameters.

For a **Bernoulli MAB (Bern-MAB)**, the conditional probability distribution $P_{r|a}$ is a Bernoulli distribution

$$P(r|a) = \mu_a^r (1 - \mu_a)^{1-r} \quad (48.4)$$

for $r \in \{0, 1\}$, where $0 < \mu_a < 1$ for each a . The parameters μ_a are initially unknown to the gambler. Note that $E[r|a] = \sum_r r P(r|a) = P(r=1|a) = \mu_a$.

For a **Gaussian MAB**, the conditional probability distribution $P_{r|a}$ is a Normal (Gaussian) distribution

$$P(r|a) = \mathcal{N}(r; \mu_a, \sigma_a^2) . \quad (48.5)$$

for $r \in \mathbb{R}$. The parameters μ_a, σ_a^2 are initially unknown to the gambler.

48.2 Reward functions

For $a \in S_{\underline{a}}$, define the **Long term Average Reward** for action a by

$$\mu_a = Q(a) = E_{|a}[\underline{r}] = \sum_r r P(r|a) . \quad (48.6)$$

Let

$$a^* = \operatorname{argmax}_a Q(a) \quad (48.7)$$

$$\mu^* = \max_a Q(a) = Q(a^*) \quad (48.8)$$

$$\Delta_a = \mu^* - Q(a) \quad (48.9)$$

μ_a, a^*, μ^* and Δ_a are not known to the gambler.

Define the **Instantaneous (at time t) Average Reward** for action a by

$$Q_t(a) = \frac{1}{N_t(a)} \sum_{\tau=0}^t r_\tau \mathbb{1}(a_\tau = a) \quad (48.10a)$$

where

$$N_t(a) = N_{in} + \sum_{\tau=0}^t \mathbb{1}(a_\tau = a) \quad (48.10b)$$

$N_{in} > 0$ insures that we never divide by zero. Note that Eqs.(48.10) can be stated recursively as

$$N_t(a)Q_t(a) = N_{t-1}(a)Q_{t-1}(a) + r_t \mathbb{1}(a_t = a) \quad (48.11a)$$

$$N_t(a) = N_{t-1}(a) + \mathbb{1}(a_t = a) \quad (48.11b)$$

with $Q_{-1}(a) = 0$ and $N_{-1}(a) = N_{in}$ for all a . We assume that at large t , $N_{in} \ll N_t(a)$ for all a . For instance, one can use $N_{in} = 1$.

$Q(a)$ is not known to the gambler but $N_t(a)$ and $Q_t(a)$ are because they are empirical.

We will write a hat over random variables that are defined by an empirical probability distribution.¹ Whereas we assume that \underline{a}_t and \underline{r}_t are i.i.d., we will not assume that \hat{a}_t and \hat{r}_t are i.i.d. for finite times t . What we will assume is that as $t \rightarrow \infty$,

¹An alternative convention is to not distinguish between \hat{a}_t and \underline{a}_t , or between \hat{r}_t and \underline{r}_t , but to distinguish between $P(r_t|a_t)$ and $\hat{P}(r_t|a_t)$, where $\hat{P}()$ is an empirical estimate of $P()$.

$$\hat{a}_t \rightarrow \underline{a}, \hat{r}_t \rightarrow \underline{r} \quad (48.12)$$

Note that

$$\sum_a \frac{N_t(a)}{t+1} = 1 \quad (48.13)$$

so define

$$P(\hat{a}_t = a) = \frac{N_t(a)}{t+1}. \quad (48.14)$$

Thus

$$E[Q(\hat{a}_t)] = \sum_a P(\hat{a}_t = a) Q(a). \quad (48.15)$$

Claim 54

$$Q_t(a) = E_{|\hat{a}_t=a}[\hat{r}_t] \quad (48.16)$$

proof:

Note that

$$\sum_r \sum_{\tau=0}^t \frac{\mathbb{1}(a_\tau = a, r_\tau = r)}{N_t(a)} = 1. \quad (48.17)$$

so define

$$P(\hat{r}_t = r | \hat{a}_t = a) = \sum_{\tau=0}^t \frac{\mathbb{1}(a_\tau = a, r_\tau = r)}{N_t(a)}. \quad (48.18)$$

Thus

$$Q_t(a) = \sum_r r \sum_{\tau=0}^t \frac{\mathbb{1}(a_\tau = a, r_\tau = r)}{N_t(a)} \quad (48.19)$$

$$= \sum_r r P(\hat{r}_t = r | \hat{a}_t = a) \quad (48.20)$$

$$= E_{|\hat{a}_t=a}[\hat{r}_t] \quad (48.21)$$

QED

Claim 55 As $t \rightarrow \infty$,

$$Q_t(a) \rightarrow E_{a=a}[\underline{r}] = Q(a) \quad (48.22)$$

$$E[Q(\hat{a}_t)] \rightarrow E[Q(\underline{a})] \quad (48.23)$$

proof: This is clear from $\hat{a}_t \rightarrow a$ and $\hat{r}_t \rightarrow r$ and Claim 54.

QED

48.3 Regret functions

Define the **Instantaneous (at time t) Average Regret (I-Regret)** by

$$IReg_t = \mu^* - E[Q(\hat{a}_t)] = E[\Delta_{\hat{a}_t}] \quad (48.24)$$

and the **Cumulative (for times $\leq t$) Average Regret (C-Regret)** by

$$CReg_t = \sum_{\tau=0}^t IReg_\tau . \quad (48.25)$$

Note that

$$CReg_t = \sum_{\tau=0}^t E[\Delta_{\hat{a}_\tau}] \quad (48.26)$$

$$= \sum_a \Delta_a \sum_{\tau=0}^t \frac{N_\tau(a)}{\tau + 1} \quad (48.27)$$

Let

$$\rho_t = \frac{1}{t+1} \sum_{t=0}^t r_t . \quad (48.28)$$

Define the **Cumulative Average Reward (C-Reward)** by

$$CRew_t = E_{|\hat{a}_t=a}[(t+1)\hat{\rho}_t] . \quad (48.29)$$

It can be shown that minimizing the C-Regret is equivalent to maximizing the C-Reward.

Claim 56 *As $t \rightarrow \infty$,*

$$IReg_t \rightarrow E[\Delta_a] \quad (48.30)$$

proof: This is clear from $\hat{a}_t \rightarrow a$ and $\hat{r}_t \rightarrow r$.

QED

48.4 Strategies with random exploration

In this section, we consider MAB algorithms that explore all values of the action a at random. In the section following this one, we consider MAB algorithm that do a more deliberate search of the action space.

48.4.1 ϵ -greedy algorithm

Recall that $\underline{a}_t \in \{0, 1, \dots, N_{\underline{a}} - 1\} = S_{\underline{a}}$. The user of the algorithm specifies an $\epsilon \in [0, 1]$ which measures the amount of exploration to be conducted.

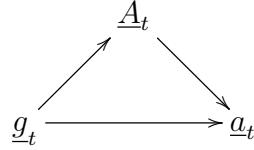


Figure 48.4: Extra structure added to MAB Bnet Fig.48.3 for ϵ -greedy algorithm.

For each t , add extra the structure shown in Fig.48.4 to the MAB bnet Fig.48.3. $g_t \in \{0, 1\}$ and $\underline{a}_t, \underline{A}_t \in S_{\underline{a}}$. (“g” stands for greedy). The TPMs, printed in blue, for the new nodes, are as follows

$$P(g_t) = \begin{cases} \epsilon \text{ (exploration)} & \text{if } g_t = 0 \\ 1 - \epsilon \text{ (exploitation)} & \text{if } g_t = 1 \end{cases}. \quad (48.31)$$

$$P(\underline{A}_t|g_t) = \begin{cases} \frac{1}{N_{\underline{a}}} \text{ (exploration)} & \text{if } g_t = 0 \\ \mathbb{1}(\underline{A}_t = 0) \text{ (exploitation)} & \text{if } g_t = 1 \end{cases} \quad (48.32)$$

$$P(\underline{a}_t|\underline{A}_t, g_t) = \begin{cases} \mathbb{1}(\underline{a}_t = \underline{A}_t) \text{ (exploration)} & \text{if } g_t = 0 \\ \mathbb{1}(\underline{a}_t = \underline{a}_t^*) \text{ (exploitation)} & \text{if } g_t = 1 \end{cases} \quad (48.33)$$

where \underline{a}_t^* is defined as follows:

$\underline{a}_t^* = \underset{a}{\operatorname{argmax}} Q_{t-1}(a)$

(48.34)

As $t \rightarrow \infty$,

$$\frac{N_t(a)}{t+1} = P(\hat{\underline{a}}_t = a) \quad (48.35)$$

$$\rightarrow P(a, g_t = 0)\epsilon + P(a, g_t = 1)(1 - \epsilon) \quad (48.36)$$

$$\geq P(a, g_t = 0)\epsilon \quad (48.37)$$

$$= \frac{\epsilon}{N_{\underline{a}}} \quad (48.38)$$

Hence

$$IReg_t = \sum_a \Delta_a \sum_{\tau=0}^t \frac{N_\tau(a)}{\tau + 1} \quad (48.39)$$

$$\geq \frac{\epsilon}{N_{\underline{a}}} \sum_a \Delta_a \quad (48.40)$$

and $CReg_t \geq \mathcal{N}(!t)(t + 1)$.

48.4.2 ϵ_t -greedy algorithm

Replace time-independent constant ϵ in the ϵ -greedy algorithm by a time dependent function ϵ_t .

48.5 Strategies with nonrandom exploration

48.5.1 Upper Confidence Bounds (UCB) algorithms

A MAB algorithm that maximizes merely $Q_t(a)$ to get a_t^* is totally greedy (i.e., does no exploration, only exploitation). This doesn't work too well because once the algo finds a particular a_t^* , it sticks with it. For times t after that, the $Q_t(a)$ for all a stay more or less the same. The $N_t(a)$ for $a \neq a_t^*$ also stay the same. Only $N_t(a_t^*)$ increases. To avoid this problem, **Upper Confidence Bounds (UCB) algorithms** maximize an effective $Q_t(a)_{\text{eff}} = Q_t(a) + U_t(a)$, where $U_t(a) > 0$, instead of maximizing merely $Q_t(a)$ to get a_t^* . $U_t(a)$ is proportional to $1/\sqrt{N_t(a)}$ (that's a property of UCBs). Adding $U_t(a)$ to $Q_t(a)$ gives those a with low $N_a(a)$ a $Q_t(a)_{\text{eff}} \gg Q_t(a)$. This encourages exploration of a different from $a_t^* = \operatorname{argmax}_a Q_t(a)$. In conclusion, for all UCB algorithms, we have

$$a_t^* = \operatorname{argmax}_a [Q_{t-1}(a) + U_{t-1}(a)] \quad (48.41)$$

Different UCB algos differ only in the definition of $U_t(a)$.

Next, we consider two UCB algorithms, frequentist UCB (UCB1), and Bayesian UCB.

Frequentist UCB (UCB1) algorithm

Claim 57 (Hoeffding's Inequality- HI) Let $\underline{x}_0, \underline{x}_1, \dots, \underline{x}_{T-1}$ be i.i.d. random variables such that $0 \leq \underline{x}_t \leq 1$ for all t . Let $\underline{m}_{T-1} = \frac{1}{T} \sum_{t=0}^{T-1} \underline{x}_t$ denote the sample mean. Then for $u > 0$, we have:

$$P(E[\underline{x}] > \underline{m}_{T-1} + u) \leq e^{-2Tu^2}. \quad (48.42)$$

proof: See Ref.[94].

QED

Let²

$$\underline{x}^T = \{\underline{x}_t : t \in \mathbb{Z}_{[0, T-1]}\} \quad (48.43)$$

and

$$\underline{r}_{\leq t-1}(a) = \{\underline{r}_\tau : \underline{a}_\tau = a, \tau \in \mathbb{Z}_{[0, t-1]}\}. \quad (48.44)$$

Note that \underline{x}^T has T components and $\underline{r}_{\leq t-1}(a)$ has $N_{t-1}(a)$ components. If we apply the HI with \underline{x}^T replaced by $\underline{r}_{\leq t-1}(a)$, we get

$$P(Q(a) > Q_{t-1}(a) + U_{t-1}(a)) \leq e^{-2N_{t-1}(a)[U_{t-1}(a)]^2} \quad (48.45)$$

If we define a threshold probability p by

$$p = e^{-2N_{t-1}(a)[U_{t-1}(a)]^2}, \quad (48.46)$$

then

$$U_{t-1}(a) = \sqrt{\frac{-\ln p}{2N_{t-1}(a)}} \quad (48.47)$$

If we choose $p = (t-1)^{-\alpha}$,

$$a_t^* = \underset{a}{\operatorname{argmax}} \left[Q_{t-1}(a) + \sqrt{\frac{\alpha(t-1)}{2N_{t-1}(a)}} \right] \quad (48.48)$$

Bayesian UCB algorithm

Claim 58 ^{3 4} (*Bayesian updating of mean and deviation. See Fig.48.5.*)
Suppose

$$x^n = [x_i], \quad x_i \text{ are i.i.d. with } \underline{x}_i | \mu, \tau \sim \mathcal{N}(\mu, \tau) \quad (48.49)$$

$$\underline{\mu} | \tau \sim \mathcal{N}(\mu_0, n_0 \tau) \quad (48.50)$$

$$\underline{\tau} \sim \text{Gamma}(\alpha, \beta). \quad (48.51)$$

Then the posterior is

²As usual, we define $\mathbb{Z}_{[a,b]} = \{a, a+1, a+2, \dots, b\}$ for $a < b$.

³For a **standard deviation** σ , the **precision** τ is defined as $\tau = \frac{1}{\sigma^2}$.

⁴ $\underline{x} | \lambda \sim \mathcal{D}(\lambda)$ means that $P(x | \lambda) = \mathcal{D}(x; \lambda)$. $\mathcal{N}()$ stands for the Normal distribution and $\text{Gamma}()$ for the Gamma distribution. See Ref.[88] for a discussion of the Gamma distribution.

$$\underline{\mu}|\tau, \underline{x}^n \sim \mathcal{N}\left(\frac{n\bar{x} + n_0\mu_0}{n + n_0}, (n + n_0)\tau\right) \quad (48.52)$$

$$\underline{\tau}|x^n \sim \text{Gamma}\left(\alpha + \frac{n}{2}, \beta + \frac{1}{2}\sum_i(x_i - \bar{x})^2 + \frac{nn_0}{2(n + n_0)}(\bar{x} - \mu_0)^2\right) \quad (48.53)$$

proof: See Ref.[17].

QED

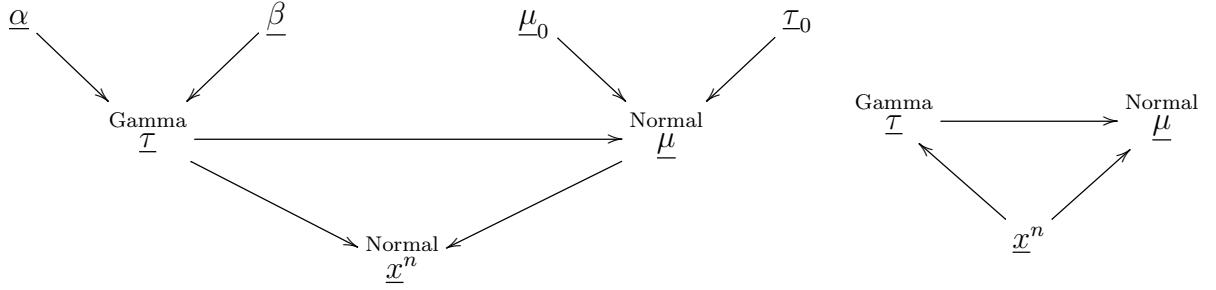


Figure 48.5: Prior and Posterior BNets in Claim 58.

In the Bayesian UCB algorithm, we use

$$a_t^* = \underset{a}{\operatorname{argmax}} E_{\mu_a, \sigma_a | s_{<t}} \left[\mu_a + c \frac{\sigma_a}{\sqrt{N_{t-1}(a)}} \right] \quad (48.54)$$

for some $c > 0$.

Eq.48.54 requires that we know $P(\mu_a, \sigma_a | s_{<t})$; i.e., the posterior distribution of μ_a, σ_a assuming the prior history $s_{<t}$. This follows from Bayes theorem if we assume a Normal distribution for the likelihood $P(s_{<t} | \mu_a, \sigma_a)$ and we assume conjugate priors for $P(\mu_a, \sigma_a)$. More precisely, if we replace \underline{x}^n by $\underline{s}_{<t}$ in Claim 58, then

$$P(\mu_a, \tau_a | s_{<t}) = P(\mu_a | \tau_a, s_{<t})P(\tau_a | s_{<t}) \quad (48.55)$$

where $P(\mu_a | \tau_a, s_{<t})$ and $P(\tau_a | s_{<t})$ are given by Claim 58.

48.5.2 Thompson Sampling MAB (TS-MAB) algorithm

Bnet for general TS-MAB algorithm

The Thompson Sampling MAB (TS-MAB) algorithm is described by the bnet Fig.48.6. This bnet differs from the bnet Fig.48.3 in that it includes new nodes λ_t . The TPMs, printed in blue, for bnet Fig.48.6, are as follows.

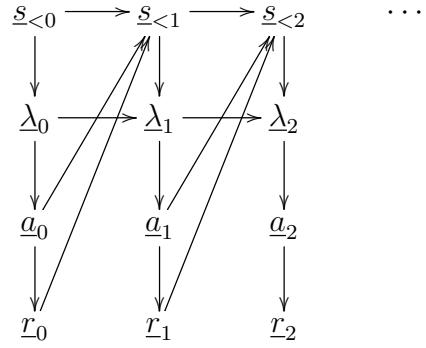


Figure 48.6: Bnet for TS-MAB algorithm.

$$P(s_{<t}|s_{<t-1}, a_{t-1}, r_{t-1}) = \text{ same as for Fig.48.3} \quad (48.56)$$

$$P(\lambda_t|s_{<t}, \lambda_{t-1}) = \mathbb{1}(\lambda_t = \lambda_t^*(s_{<t}, \lambda_{t-1})) \quad (48.57)$$

where λ_t^* is a function to be defined below.

$$P(a_t|\lambda_t) = \mathbb{1}[a_t = a_t(\lambda_t)] \quad (\text{Agent's response}) \quad (48.58)$$

where $a_t(\lambda_t)$ is a function to be described below.

$$P(r_t|a_t) = \text{ same as for Fig.48.3. (Environment's response)} \quad (48.59)$$

Let

$$Q_t(a, \lambda_t) = \sum_r r P(\hat{r}_t = r | \hat{a}_t = a, \lambda_t) \quad (48.60)$$

$$= E_{\hat{a}_t=a, \lambda_t} [\hat{r}_t]. \quad (48.61)$$

Define

$$a_t(\lambda_t) = \operatorname{argmax}_a Q_t(a, \lambda_t).$$

(48.62)

Note that

$$P(a_t|s_{<t}) = \sum_{\lambda_t} P(\lambda_t|s_{<t}) \mathbb{1}[Q_t(a_t, \lambda_t) = \max_a Q_t(a, \lambda_t)] \quad (48.63a)$$

$$= \sum_{\lambda_t} P(\lambda_t|s_{<t}) \mathbb{1}[a_t = a_t(\lambda_t)] \quad (48.63b)$$

$$= E_{\lambda_t|s_{<t}} \{ \mathbb{1}[a_t = a_t(\lambda_t)] \}. \quad (48.63c)$$

If we further assume that $P(\lambda_t|s_{<t})$ is a delta function, then Eqs.(48.63) reduce to

$$P(a_t|s_{<t}) = \mathbb{1}[a_t = a_t(\lambda_t^*)] \quad (48.64)$$

TS-MAB algorithm with Beta agent and Bernoulli environment

The Beta distribution $\text{Beta}(x; \alpha, \beta)$ (see Ref.[72]) is defined for $\alpha > 0, \beta > 0$ and $x \in [0, 1]$. Since $x \in [0, 1]$, x can be interpreted as a probability. The mean and variance of the Beta distribution are

$$E[\underline{x}] = \frac{\alpha}{\alpha + \beta}, \quad (48.65)$$

$$\langle \underline{x}, \underline{x} \rangle = \frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)}. \quad (48.66)$$

From this mean and variance, we see that if we increase α by one and leave β the same, the mean moves towards 1 and the variance decreases. Likewise, if we increase β by 1 and leave α the same, the mean moves towards 0 and the variance decreases.

Let

$$\alpha_t^a = \sum_{\tau=0}^t \mathbb{1}(r_\tau = 1, a_\tau = a) \quad (48.67)$$

$$\beta_t^a = \sum_{\tau=0}^t \mathbb{1}(r_\tau = 0, a_\tau = a) \quad (48.68)$$

$$\lambda_t^a = (\alpha_{t-1}^a, \beta_{t-1}^a) \quad (48.69)$$

$$\lambda_t = [(\alpha_{t-1}^a, \beta_{t-1}^a) : a \in S_a] \quad (48.70)$$

$$Q_t(a, \lambda_t) = \sum_r r \text{Beta}(r; \alpha_{t-1}^a, \beta_{t-1}^a) \quad (48.71a)$$

$$= \frac{\alpha_{t-1}^a}{\alpha_{t-1}^a + \beta_{t-1}^a} \quad (48.71b)$$

The TS-MAB algorithm for a Beta agent and Bernoulli environment can be described by the bnet Fig.48.6. For this special case of bnet Fig.48.6, the TPMs, printed in blue, are as follows.

$$P(s_{<t}|s_{<t-1}, a_{t-1}, r_{t-1}) = \text{ same as for Fig.48.6} \quad (48.72)$$

$$P(\lambda_t^a|s_{<t}, \lambda_{t-1}) = \mathbb{1}(\lambda_t^a = \lambda_{t-1}^a) = \begin{cases} \mathbb{1}(\lambda_t^a = \lambda_{t-1}^a) & \text{if } a \neq a_{t-1} \\ \mathbb{1}(\lambda_t^a = (\alpha_{t-2}^a + 1, \beta_{t-2}^a)) & \text{if } a_{t-1} = a \text{ and } r_{t-1} = 1 \\ \mathbb{1}(\lambda_t^a = (\alpha_{t-2}^a, \beta_{t-2}^a + 1)) & \text{if } a_{t-1} = a \text{ and } r_{t-1} = 0 \end{cases} \quad (48.73)$$

$$P(a_t|\lambda_t) = \mathbb{1}(a_t = a_t(\lambda_t)) \quad (48.74)$$

$$= \mathbb{1}(a_t = \operatorname{argmax}_a \underbrace{Q_t(a, \lambda_t)}_{\text{see Eq.(48.71)}}) \quad (\text{Beta agent response}) \quad (48.75)$$

$$P(r_t|a_t) = \mu_{a_t}^{r_t} (1 - \mu_{a_t})^{r_t} \quad (\text{Bernoulli environment response}) . \quad (48.76)$$

μ_a known to environment but unknown to agent.

TS-MAB algorithm, skeletal reprise

The TS-MAB algorithm is not very complicated but explaining it with precision requires nightmarishly many indices. Here is a pedagogical reprise of what we have said so far, where we have stripped out some of the inessential indices.

$$P(\lambda) = \mathbb{1}(\lambda, \lambda^*) \quad (48.77)$$

$$P(a|\lambda) = \mathbb{1}(a = a(\lambda)) \quad (48.78)$$

$$= \mathbb{1}(a = \operatorname{argmax}_a \sum_r r P(\hat{r} = r | \hat{a} = a, \lambda)) \quad (48.79)$$

$$= \mathbb{1}(a = \operatorname{argmax}_a \sum_r r \text{Beta}(r; \lambda^a)) \quad (\text{Beta agent response}) \quad (48.80)$$

Define $q()$ by

$$q(r; \lambda^a) = P(\hat{r} = r | \hat{a} = a, \lambda) \quad (48.81)$$

- PRIOR:

$$P(a) = \sum_{\lambda} P(\lambda)P(a|\lambda) \quad (48.82)$$

$$= P(a|\lambda^*) \quad (48.83)$$

$$= \mathbb{1}(a = \operatorname{argmax}_a \sum_r r q(r; \lambda^{*a})) \quad (48.84)$$

1. Use fact that $q = \text{Beta}$

$$\sum_r r \text{Beta}(r; \lambda^{*a}) = \frac{\alpha^{*a}}{\alpha^{*a} + \beta^{*a}} \quad (48.85)$$

2. Don't use fact that $q = \text{Beta}$.

For each a , get samples $r^\sigma \sim q(r; \lambda^{*a})$ for $\sigma = 0, 1, \dots, nsam - 1$ and estimate

$$\sum_r r q(r; \lambda^{*a}) \approx \frac{1}{nsam} \sum_{\sigma} r^\sigma q(r^\sigma; \lambda^{*a}). \quad (48.86)$$

This sampling is why TS is called a sampling.

- LIKELIHOOD:

$$P(r|a) = \mu_a^r (1 - \mu_a)^{r'} \quad (\text{Bernoulli environment response}) \quad (48.87)$$

48.5.3 Grad-MAB algorithm

Let

$$\lambda_{t+1}(a) = \lambda_t(a) + \eta r_t [\mathbb{1}(a_t = a) - \pi_t(a)] \quad (48.88)$$

for some $\eta > 0$, where $\pi_t(a)$ is defined by

$$\pi_t(a) = P(\underline{a}_t = a | \lambda_t) = \underbrace{\frac{e^{\lambda_t(a)}}{\sum_a e^{\lambda_t(a)}}}_{\text{softmax}(\lambda_t)_a}. \quad (48.89)$$

The $\lambda_t(a)$ are called **scores** at time t . Let

$a_t^* = \operatorname{argmax}_a \pi_t(a)$

(48.90)

The Gradient MAB (Grad-MAB) algorithm is described by the bnet Fig.48.7. This bnet differs from the bnet Fig.48.3 in that it includes new nodes $\underline{\lambda}_t$. The TPMs, printed in blue, for bnet Fig.48.7, are as follows.

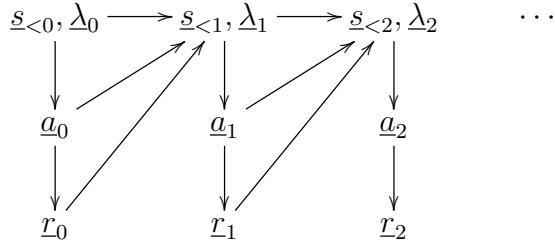


Figure 48.7: Bnet for Grad-MAB algorithm.

$$P(\underline{\lambda}_{t+1} = \lambda | a_t, r_t, \underline{\lambda}_t) = \mathbb{1}(\lambda = \lambda_{t+1} \text{ given by Eq.(48.88)}). \quad (48.91)$$

$$P(a_t | \lambda_t) = \mathbb{1}(a_t = a_t^*) \quad \text{or, alternatively, } = P(\underline{a}_t = a_t | \lambda_t) = \pi_t(a_t) \quad (\text{agent's response}) \quad (48.92)$$

$$P(r_t | a_t) = P_{\underline{r}|\underline{a}}(r_t | a_t) \quad (\text{environment's response}) . \quad (48.93)$$

Motivation:

Define the **Instantaneous Average Reward** by

$$\mathcal{R}_t(\lambda_t) = \sum_a P(\underline{a}_t = a | \lambda_t) E_{r_t | a_t = a}[\underline{r}_t] = E_{r_t, \underline{a}_t | \lambda_t}[\underline{r}_t] . \quad (48.94)$$

We will assume that as $t \rightarrow \infty$, $\underline{a}_t \rightarrow \underline{a}$, $\underline{r}_t \rightarrow \underline{r}$ and $\lambda_t \rightarrow \lambda$. Therefore, $\mathcal{R}_t(\lambda_t) \rightarrow E_{\underline{r}, \underline{a} | \lambda}[\underline{r}] = E_{\underline{r} | \lambda}[\underline{r}]$.

Note that if $E_{r_t | a_t = a}[\underline{r}_t] = B$, where B is independent of a , then $\mathcal{R}_t(\lambda_t) = B$ and $\frac{\partial \mathcal{R}_t}{\partial \lambda_t(a)} = 0$.

Claim 59 *The gradient of $\mathcal{R}_t(\lambda_t)$ is*

$$\frac{\partial \mathcal{R}_t}{\partial \lambda_t(a)} = E_{r_t, \underline{a}_t | \lambda_t}[g(r_t, \underline{a}_t | a, \lambda_t)] \quad (48.95)$$

where

$$g(r_t, \underline{a}_t | a, \lambda_t) = \underline{r}_t [\mathbb{1}(\underline{a}_t = a) - \pi_t(a)] \quad (48.96)$$

proof:

$$\frac{\partial \mathcal{R}_t}{\partial \lambda_t(a)} = \sum_{a'} \frac{\partial \pi_t(a')}{\partial \lambda_t(a)} E_{\underline{r}_t | \underline{a}_t = a'} [\underline{r}_t] \quad (48.97)$$

$$= \sum_{a'} \pi_t(a') \frac{\partial \ln \pi_t(a')}{\partial \lambda_t(a)} E_{\underline{r}_t | \underline{a}_t = a'} [\underline{r}_t] \quad (48.98)$$

$$= \sum_{a'} \pi_t(a') \frac{\partial}{\partial \lambda_t(a)} \left[\ln \frac{\exp[\lambda_t(a')]}{\sum_a \exp[\lambda_t(a)]} \right] E_{\underline{r}_t | \underline{a}_t = a'} [\underline{r}_t] \quad (48.99)$$

$$= \sum_{a'} \pi_t(a') [\mathbb{1}(a' = a) - \pi_t(a)] E_{\underline{r}_t | \underline{a}_t = a'} [\underline{r}_t] \quad (48.100)$$

$$= \sum_{a'} P(\underline{a}_t = a' | \lambda_t) E_{\underline{r}_t | \underline{a}_t = a'} [\underline{r}_t (\mathbb{1}(a' = a) - \pi_t(a))] \quad (48.101)$$

$$= E_{\underline{r}_t, \underline{a}_t | \lambda_t} [g(\underline{r}_t, \underline{a}_t | a, \lambda_t)] \quad (48.102)$$

QED

Eq.(48.88) can be written as

$$\lambda_{t+1}(a) = \lambda_t(a) + \eta g(r_t, a_t | a, \lambda_t) \quad (48.103)$$

Chapter 49

Naive Bayes

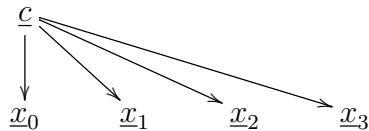


Figure 49.1: bnet for Naive Bayes with 4 features

Class node $\underline{c} \in S_{\underline{c}}$. $|S_{\underline{c}}| = n_{\underline{c}}$ = number of classes.

Feature nodes $\underline{x}_i \in S_{\underline{x}_i}$ for $i = 0, 1, 2, \dots, F - 1$. F =number of features.

Define

$$x. = [x_0, x_1, \dots, x_{F-1}] . \quad (49.1)$$

For the bnet of Fig.49.1,

$$P(c, x.) = P(c) \prod_{i=0}^{F-1} P(x_i|c) . \quad (49.2)$$

Given $x.$ values, find most likely class $c \in S_{\underline{c}}$.

Maximum a Posteriori (MAP) estimate:

$$c^* = \operatorname{argmax}_c P(c|x.) \quad (49.3)$$

$$= \operatorname{argmax}_c \frac{P(c, x.)}{P(x.)} \quad (49.4)$$

$$= \operatorname{argmax}_c P(c, x.) . \quad (49.5)$$

Chapter 50

Neural Networks

In this chapter, we discuss Neural Networks (NNs) of the feedforward kind, which is the most popular kind. In their plain, vanilla form, NNs only have deterministic nodes. But the nodes of a bnet can be deterministic too, because the TPM of a node can reduce to a delta function. Hence, NNs should be expressible as bnets. We will confirm this in this chapter.

Henceforth in this chapter, if we replace an index of an indexed quantity by a dot, it will mean the collection of the indexed quantity for all values of that index. For example, $\underline{x}.$ will mean the array of x_i for all i .

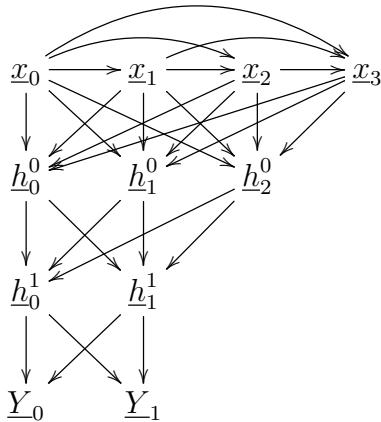


Figure 50.1: Neural Network (feed forward) with 4 layers: input layer $\underline{x}.$, 2 hidden layers $\underline{h}^0.$, $\underline{h}^1.$ and output layer $\underline{Y}.$

Consider Fig.50.1.

$\underline{x}_i \in \{0, 1\}$ for $i = 0, 1, 2, \dots, nx - 1$ is the **input layer**.

$\underline{h}_i^\lambda \in \mathbb{R}$ for $i = 0, 1, 2, \dots, nh(\lambda) - 1$ is the λ -th **hidden layer**. $\lambda = 0, 1, 2, \dots, \Lambda - 2$. A NN is said to be **deep** if $\Lambda > 2$; i.e., if it has more than one hidden layer.

$\underline{Y}_i \in \mathbb{R}$ for $i = 0, 1, 2, \dots, ny - 1$ is the **output layer**. We use a upper case y here because in the training phase, we will use pairs $(x.[\sigma], y.[\sigma])$ where $y_i[\sigma] \in \{0, 1\}$

for $i = 0, 1, \dots, ny - 1$. $Y = \hat{y}$ is an estimate of y . Note that lower case y is either 0 or 1, but upper case Y may be any real. Often, the activation functions are chosen so that $Y \in [0, 1]$.

The number of nodes in each layer and the number of layers are arbitrary. Fig.50.1 is fully connected (aka dense), meaning that every node of a layer is impinged arrow coming from every node of the preceding layer. Later on in this chapter, we will discuss non-dense layers.

Let $w_{i|j}^\lambda, b_i^\lambda \in \mathbb{R}$ be given, for $i \in \mathbb{Z}_{[0, nh(\lambda)]}$, $j \in \mathbb{Z}_{[0, nh(\lambda-1)]}$, and $\lambda \in \mathbb{Z}_{[0, \Lambda]}$.

The TPMs, printed in blue, for bnet Fig.50.1, are as follows.

$$P(x_i | x_{i-1}, x_{i-2}, \dots, x_0) = \text{given} \quad (50.1)$$

$$P(h_i^\lambda | h_{\cdot}^{\lambda-1}) = \delta \left(h_i^\lambda, \mathcal{A}_i^\lambda \left(\sum_j w_{i|j}^\lambda h_j^{\lambda-1} + b_i^\lambda \right) \right), \quad (50.2)$$

where $P(h_i^0 | h^{-1}) = P(h_i^0 | x)$.

$$P(Y_i | h_{\cdot}^{\lambda-2}) = \delta \left(Y_i, \mathcal{A}_i^{\lambda-1} \left(\sum_j w_{i|j}^{\lambda-1} h_j^{\lambda-2} + b_i^{\lambda-1} \right) \right). \quad (50.3)$$

50.1 Activation Functions $\mathcal{A}_i^\lambda : \mathbb{R} \rightarrow \mathbb{R}$

Activation functions must be nonlinear. Why? Because if they were all linear, the NN mapping would be a bijection (1-1 onto map), and its domain and range would be the same. That is not what you want for a classifier. For a classifier, you want the range to be much smaller than the domain.

- **Step function (Perceptron)**

$$\mathcal{A}(x) = \mathbb{1}(x > 0) \quad (50.4)$$

Zero for $x \leq 0$, one for $x > 0$.

- **Sigmoid function**

$$\mathcal{A}(x) = \frac{1}{1 + e^{-x}} = \text{smoid}(x) \quad (50.5)$$

Smooth, monotonically increasing function. $\text{smoid}(-\infty) = 0, \text{smoid}(0) = 1/2, \text{smoid}(\infty) = 1$.

- **Hyperbolic tangent**

$$\mathcal{A}(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (50.6)$$

Smooth, monotonically increasing function. $\tanh(-\infty) = -1$, $\tanh(0) = 0$, $\tanh(\infty) = 1$.

Odd function:

$$\tanh(-x) = -\tanh(x) \quad (50.7)$$

Whereas smoid(x) $\in [0, 1]$, $\tanh(x) \in [-1, 1]$.

- **ReLU (Rectified Linear Unit)**

$$\mathcal{A}(x) = \underbrace{x \mathbb{1}(x > 0)}_{x_+} = \max(0, x) . \quad (50.8)$$

Compare this to the step function $\mathbb{1}(x > 0)$.

- **Swish**

$$\mathcal{A}(x) = x \text{ smoid}(x) \quad (50.9)$$

- **Softmax**

$$\mathcal{A}(x_i|x.) = \frac{e^{x_i}}{\sum_i e^{x_i}} = \text{softmax}(x.)_i \quad (50.10)$$

The softmax definition implies that the bnet nodes within a softmax layer are fully connected by arrows to form a “clique”.

50.2 Weight optimization via supervised training and gradient descent

The bnet of Fig.50.1 is used for classification of a single data point $x..$. It assumes that the weights $w_{ij}^\lambda, b_i^\lambda$ are given.

To find the optimum weights via supervised training and gradient descent, one uses the bnet Fig.50.2.

In Fig.50.2, the nodes in Fig.50.1 become sampling space vectors. For example, $\underline{x}..$ becomes $\vec{x}_..$, where the components of $\vec{x}_..$ in sampling space are $\underline{x}..[\sigma] \in \{0, 1\}^{nx}$ for $\sigma = 0, 1, \dots, nsam(\vec{x}) - 1$.

$nsam(\vec{x})$ is the number of samples used to calculate the gradient during each **stage (aka iteration)** of Fig.50.2. We will also refer to $nsam(\vec{x})$ as the **mini-batch size**. A **mini-batch** is a subset of the training dataset.

To train a bnet with a data set (d-set), the standard procedure is to split the d-set into 3 parts:

1. **training d-set**,
2. **testing1 d-set**, for tuning of hyperparameters like $nsam(\vec{x})$, Λ , and $nh(i)$ for each i .
3. **testing2 d-set**, for measuring how well the model tuned with the testing1 d-set performs.

The training d-set is itself split into mini-batches. An **epoch** is a pass through all the training d-set.

Define

$$W_{i|j}^\lambda = [w_{i|j}^\lambda, b_i^\lambda] . \quad (50.11)$$

The TPMs, printed in blue, for bnet Fig.50.2, are as follows.

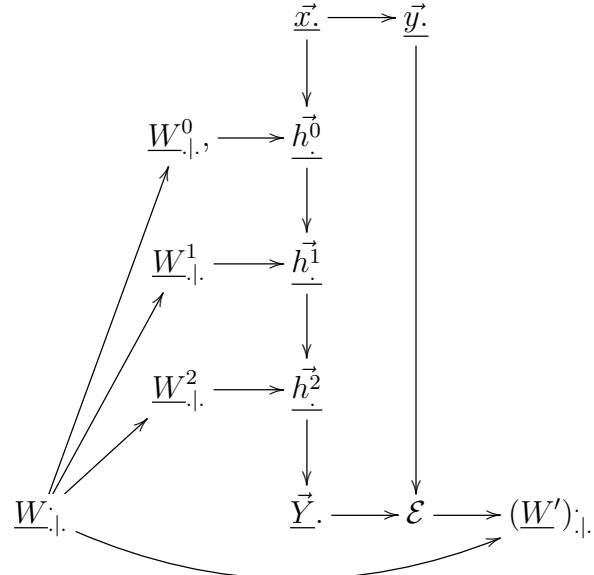


Figure 50.2: bnet for finding optimum weights of the bnet Fig.50.1 via supervised training and gradient descent.

$$P(x.[\sigma]) = \text{given} . \quad (50.12)$$

$$P(y.[\sigma] | x.[\sigma]) = \text{given} . \quad (50.13)$$

$$P(h_i^\lambda[\sigma] | h_j^{\lambda-1}[\sigma]) = \delta \left(h_i^\lambda[\sigma], \mathcal{A}_i^\lambda \left(\sum_j w_{i|j}^\lambda h_j^{\lambda-1}[\sigma] + b_i^\lambda \right) \right) \quad (50.14)$$

$$P(Y_i[\sigma] | h_{\cdot}^{\Lambda-2}[\sigma]) = \delta \left(Y_i[\sigma], \mathcal{A}_i^{\Lambda-1} \left(\sum_j w_{i|j}^{\Lambda-1} h_j^{\Lambda-2}[\sigma] + b_i^{\Lambda-1} \right) \right) \quad (50.15)$$

$$P(W_{\cdot|\cdot}) = \text{given} \quad (50.16)$$

The first time it is used, $W_{\cdot|\cdot}$ is arbitrary. After the first time, it is determined by previous stage.

$$P(W_{\cdot|\cdot}^{\lambda} | W_{\cdot|\cdot}) = \delta(W_{\cdot|\cdot}^{\lambda}, (W_{\cdot|\cdot})^{\lambda}) \quad (50.17)$$

$$P(\mathcal{E} | \vec{y}_{\cdot}, \vec{Y}_{\cdot}) = \frac{1}{nsam(\vec{x})} \sum_{\sigma} \sum_i d(y_i[\sigma], Y_i[\sigma]) , \quad (50.18)$$

where

$$d(y, Y) = |y - Y|^2 . \quad (50.19)$$

If $y, Y \in [0, 1]$, one can use this instead

$$d(y, Y) = XE(y \rightarrow Y) = -y \ln Y - (1 - y) \ln(1 - Y) . \quad (50.20)$$

$$P((W')_{i|j}^{\lambda} | \mathcal{E}, W_{\cdot|\cdot}) = \delta((W')_{i|j}^{\lambda}, W_{i|j}^{\lambda} - \eta \partial_{W_{i|j}^{\lambda}} \mathcal{E}) \quad (50.21)$$

$\eta > 0$ is called the learning rate. This method of minimizing the error \mathcal{E} is called gradient descent. $W' - W = \Delta W = -\eta \partial_W \mathcal{E}$ so $\Delta \mathcal{E} = \frac{-1}{\eta} (\Delta W)^2 < 0$.

50.3 Non-dense layers

The TPM for a non-dense layer is of the form:

$$P(h_i^{\lambda}[\sigma] | h_{\cdot}^{\lambda-1}[\sigma]) = \delta(h_i^{\lambda}[\sigma], H_i^{\lambda}[\sigma]) , \quad (50.22)$$

where $H_i^{\lambda}[\sigma]$ will be specified below for each type of non-dense layer.

- **Dropout Layer**

The dropout layer was invented in Ref.[51]. To dropout nodes from a fixed layer λ : For all i of layer λ , define a new node $\underline{r}_i^{\lambda}$ with an arrow $\underline{r}_i^{\lambda} \rightarrow \underline{h}_i^{\lambda}$. For $r \in \{0, 1\}$, and some $p \in (0, 1)$, define

$$P(r_i^\lambda = r) = [p]^r [1-p]^{1-r} \text{ (Bernoulli dist.) .} \quad (50.23)$$

Now one has

$$P(h_i^\lambda[\sigma] | h_j^{\lambda-1}[\sigma], r_i^\lambda) = \delta(h_i^\lambda[\sigma], H_i^\lambda[\sigma]) , \quad (50.24)$$

where

$$H_i^\lambda[\sigma] = \mathcal{A}_i^\lambda(r_i^\lambda \sum_j w_{i|j}^\lambda h_j^{\lambda-1}[\sigma] + b_i^\lambda) . \quad (50.25)$$

This reduces overfitting. Overfitting might occur if the weights follow too closely several similar minibatches. This dropout procedure adds a random component to each minibatch making groups of similar minibatches less likely.

The random r_i^λ nodes that induce dropout are only used in the training bnet Fig.50.2, not in the classification bnet Fig.50.1. We prefer to remove the r_i^λ stochasticity from classification and for Fig.50.1 to act as an average over sampling space of Fig.50.2. Therefore, if weights $w_{i|j}^\lambda$ are obtained for a dropout layer λ in Fig.50.2, then that layer is used in Fig.50.1 with no r_i^λ nodes but with weights $\langle r_i^\lambda \rangle w_{i|j}^\lambda = pw_{i|j}^\lambda$.

Note that dropout adds non-deterministic nodes to a NN, which in their vanilla form only have deterministic nodes.

- **Convolutional Layer**

- **1-dim**

Filter function $\mathcal{F} : \{0, 1, \dots, nf - 1\} \rightarrow \mathbb{R}$.

σ =stride length

For $i \in \{0, 1, \dots, nh(\lambda) - 1\}$, let

$$H_i^\lambda[\sigma] = \sum_{j=0}^{nf-1} h_{j+i\sigma}^{\lambda-1}[\sigma] \mathcal{F}(j) . \quad (50.26)$$

For the indices not to go out of bounds in Eq.(50.26), we must have

$$nh(\lambda - 1) - 1 = nf - 1 + (nh(\lambda) - 1)\sigma \quad (50.27)$$

so

$$nh(\lambda) = \frac{1}{\sigma}[nh(\lambda - 1) - nf] + 1 . \quad (50.28)$$

- **2-dim**

$h_i^\lambda[\sigma]$ becomes $h_{(i,j)}^\lambda[\sigma]$. Do 1-dim convolution along both i and j axes.

- **Pooling Layers (MaxPool, AvgPool)**

Here each node i of layer λ is impinged by arrows from a subset $Pool(i)$ of the set of all nodes of the previous layer $\lambda - 1$. Partition set $\{0, , 1, \dots, nh(\lambda - 1) - 1\}$ into $nh(\lambda)$ mutually disjoint, nonempty sets called $Pool(i)$, where $i \in \{0, 1, \dots, nh(\lambda) - 1\}$.

- AvgPool

$$H_i^\lambda[\sigma] = \frac{1}{|Pool(i)|} \sum_{j \in Pool(i)} h_j^{\lambda-1}[\sigma] \quad (50.29)$$

- MaxPool

$$H_i^\lambda[\sigma] = \max_{j \in Pool(i)} h_j^{\lambda-1}[\sigma] \quad (50.30)$$

50.4 Autoencoder NN

If the sequence

$$nx, nh(0), nh(1), \dots, nh(\Lambda - 2), ny \quad (50.31)$$

first decreases monotonically up to layer λ_{min} , then increases monotonically until $ny = nx$, then the NN is called an **autoencoder NN**. Autoencoders are useful for unsupervised learning and feature reduction. In this case, Y estimates x . The layers before layer λ_{min} are called the **encoder**, and those after λ_{min} are called the **decoder**. Layer λ_{min} is called the **code**.

Chapter 51

Noisy-OR gate

The Noisy-OR gate was first proposed by Judea Pearl in his 1988 book Ref.[37].

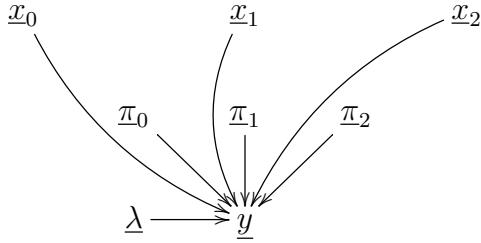


Figure 51.1: Noisy-OR gate $\underline{y} \in \{0, 1\}$ with $n = 3$, Boolean inputs $(\underline{x}_i)_{i=0,1,2}$ and parameters $\underline{\lambda}, (\underline{\pi})_{i=0,1,2}$.

Let

$\underline{\lambda} \in [0, 1]$ =gate leakage.

$y \in \{0, 1\}$ = gate output

$\underline{x}^n = (\underline{x}_i)_{i=0,1,\dots,n-1}$, where $\underline{x}_i \in \{0, 1\}$ are gate inputs.

$\underline{\pi}^n = (\underline{\pi}_i)_{i=0,1,\dots,n-1}$, where $\underline{\pi}_i \in [0, 1]$ are gate parameters.

The TPM, printed in blue, for the Noisy-OR gate \underline{y} shown in Fig.51.1, is as follows.

$$P(y = 1 | \underline{x}^n, \underline{\lambda}, \underline{\pi}^n) = 1 - (1 - \lambda) \prod_i [1 - \pi_i x_i] \quad (51.1)$$

$$P(y = 0 | \underline{x}^n, \underline{\lambda}, \underline{\pi}^n) = 1 - P(y = 1 | \underline{x}^n, \underline{\lambda}, \underline{\pi}^n) \quad (51.2)$$

Note that if $\lambda = 0$ and $\pi_i = 1$ for all i , then this becomes a deterministic OR-gate. Indeed,

$$P(y = 1 | \underline{x}^n, \lambda = 0, \underline{\pi}^n = 1^n) = 1 - \prod_i [1 - x_i] = \vee_{i=0}^{n-1} x_i , \quad (51.3)$$

so

$$P(y|x^n, \lambda = 0, \pi^n = 1^n) = \delta(y, \vee_{i=0}^{n-1} x_i) . \quad (51.4)$$

51.1 3 ways to interpret the parameters π_i

1. Note that if $\lambda = 0$ and x^n is one hot (i.e., $x^n = e_i^n$, where e_i^n is the vector with all components zero except for the i -th component which equals 1), then

$$P(y = 1|x^n = e_i^n, \lambda = 0, \pi^n) = 1 - [1 - \pi_i] = \pi_i . \quad (51.5)$$

This gives an interpretation to the parameters π_i .

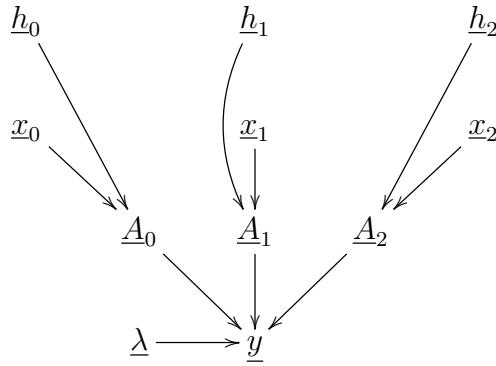


Figure 51.2: Fig.51.1 after replacing parameters $(\pi_i)_{i=0,1,2}$ by hidden nodes $(h_i)_{i=0,1,2}$.

2. Another way of interpreting the parameters π_i is to associate each of them with a hidden variable $h_i \in \{0, 1\}$ whose average equals π_i . More precisely, consider Fig.51.2.

Let $x_i, h_i, A_i, y \in \{0, 1\}$.

The TPMs, printed in blue, for the bnet Fig.51.2, are as follows:

$$P(h_i) = \pi_i \delta(h_i, 1) + (1 - \pi_i) \delta(h_i, 0) \quad (51.6)$$

$$P(A_i|h_i, x_i) = \delta(A_i, h_i \wedge x_i) = \delta(A_i, h_i x_i) \quad (51.7)$$

$$P(y = 1|A^n) = 1 - (1 - \lambda) \wedge_{i=0}^{n-1} \bar{A}_i \quad (51.8)$$

$$= 1 - (1 - \lambda) \prod_i (1 - A_i) \quad (51.9)$$

$$P(y = 0|A^n) = 1 - P(y = 1|A^n) \quad (51.10)$$

Note that

$$P(y = 1|x^n, \lambda) = \sum_{h^n} \sum_{A^n} \left[1 - (1 - \lambda) \prod_i (1 - A_i) \right] [\prod_i \delta(A_i, h_i x_i)] P(h^n) \quad (51.11)$$

$$= E_{\underline{h}^n} \left[[1 - (1 - \lambda) \prod_i (1 - h_i x_i)] \right]. \quad (51.12)$$

But

$$E_{\underline{h}_i}[h_i x_i] = \sum_{h_i=0,1} P(h_i) h_i x_i = \pi_i x_i \quad (51.13)$$

so

$$P(y = 1|x^n, \lambda) = 1 - (1 - \lambda) \prod_i (1 - \pi_i x_i). \quad (51.14)$$

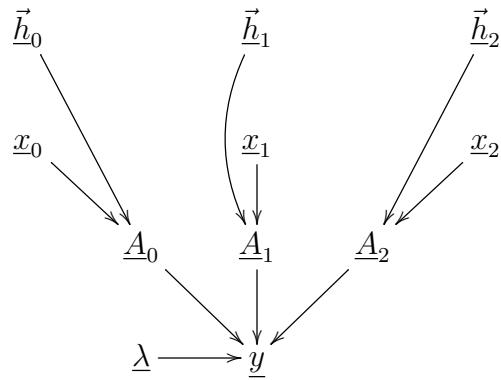


Figure 51.3: Fig.51.2 after replacing the hidden nodes $(\underline{h}_i)_{i=0,1,2}$ by vectors of samples $(\vec{h}_i)_{i=0,1,2}$.

3. Another way to interpret the parameters π_i is to associate each of them with a vector of samples \vec{h}_i whose average is π_i . More precisely, consider Fig.51.3.

Suppose $\underline{h}_i \in \{0, 1\}$ and define

$$P_{\underline{h}_i}(h_i) = \pi_i \delta(h_i, 1) + (1 - \pi_i) \delta(h_i, 0) . \quad (51.15)$$

Suppose $\vec{h}_i = (\underline{h}_i[\sigma])_{s=0,1,\dots,n_{sam}-1}$ and the Boolean samples $\underline{h}_i[\sigma] \in \{0, 1\}$ are i.i.d. with $\underline{h}_i[\sigma] \sim P_{\underline{h}_i}$ for all σ .

Note that for each i , an estimate $\hat{P}_{\underline{h}_i}(h_i)$ of $P_{\underline{h}_i}(h_i)$ can be obtained from the vector of samples \vec{h}_i as follows:

$$\hat{P}_{\underline{h}_i}(h_i) = \frac{1}{n_{sam}} \sum_{\sigma=0}^{n_{sam}-1} \mathbb{1}(\underline{h}_i[\sigma] = h_i) . \quad (51.16)$$

Let $x_i, \underline{h}_i[\sigma], \underline{A}_i, y \in \{0, 1\}$.

The TPMs, printed in blue, for the bnet Fig.51.3, are as follows:

$$P(\vec{h}_i) = \prod_{\sigma=0}^{n_{sam}-1} P_{\underline{h}_i}(\underline{h}_i[\sigma]) \quad (51.17)$$

$$P(A_i | \vec{h}_i, x_i) = \delta(A_i, \frac{1}{n_{sam}} \sum_{\sigma} \underline{h}_i[\sigma] \wedge x_i) \quad (51.18)$$

$$= \delta(A_i, \pi_i x_i) \quad (51.19)$$

$$P(y = 1 | A^n) = 1 - (1 - \lambda) \wedge_{i=0}^{n-1} \bar{A}_i \quad (51.20)$$

$$= 1 - (1 - \lambda) \prod_i (1 - A_i) \quad (51.21)$$

$$P(y = 0 | A^n) = 1 - P(y = 1 | A^n) \quad (51.22)$$

Note that

$$P(y = 1 | x^n, \lambda, \vec{h}^n) = \sum_{A^n} \left[1 - (1 - \lambda) \prod_i (1 - A_i) \right] \prod_i \delta(A_i, \pi_i x_i) \quad (51.23)$$

$$= 1 - (1 - \lambda) \prod_i (1 - \pi_i x_i) . \quad (51.24)$$

Chapter 52

Non-negative Matrix Factorization

Based on Ref.[117].

Given matrix V , factor it into product of two matrices

$$V = WH , \quad (52.1)$$

where all 3 matrices have non-negative entries.

$V \in \mathbb{R}_{\geq 0}^{nv \times na}$: visible info matrix

$W \in \mathbb{R}_{\geq 0}^{nv \times nh}$: weight info matrix

$H \in \mathbb{R}_{\geq 0}^{nh \times na}$: hidden info matrix

Usually, $nv > nh < na$ so compression of information (aka dimensional reduction, clustering)

52.1 Bnet interpretation

Express node \underline{v} as a chain of two nodes.

$$\underline{v} \longleftarrow \underline{a} \quad = \quad \underline{w} \longleftarrow \underline{h} \longleftarrow \underline{a}$$

Figure 52.1: Bnet interpretation of non-negative matrix factorization.

The TPMs, printed in blue, for bnet Fig.52.1, are as follows.

$$P(\underline{v} = w | \underline{a}) = \frac{V_{w,a}}{\sum_w V_{w,a}} \quad (52.2)$$

$$P(w|h) = \frac{W_{w,h}}{\sum_w W_{w,h}} \quad (52.3)$$

$$P(h|a) = \frac{\sum_w W_{w,h}}{\sum_w V_{w,a}} H_{h,a} \quad (52.4)$$

52.2 Simplest recursive algorithm

Initialize: Choose nh . Choose $W^{(0)}$ and $H^{(0)}$ that have non-negative entries.

Update: For $n = 0, 1, \dots$, do

$$H_{i,j}^{(n+1)} \leftarrow H_{i,j}^{(n)} \frac{[(W^{(n)})^T V]_{i,j}}{[(W^{(n)})^T \underbrace{W^{(n)} H^{(n)}}_{\approx V}]_{i,j}} \quad (52.5)$$

and

$$W_{i,j}^{(n+1)} \leftarrow W_{i,j}^{(n)} \frac{[V(H^{(n+1)})^T]_{i,j}}{\underbrace{[W^{(n)} H^{(n+1)} (H^{(n+1)})^T]_{i,j}}_{\approx V}}. \quad (52.6)$$

After each step, record error defined by

$$\mathcal{E}^{(n)} = \| V - W^{(n)} H^{(n)} \|_2. \quad (52.7)$$

Using 2-norm, aka Frobenius matrix norm. Continue until reach acceptable error.

Can also use Kullback-Liebler divergence for error:

$$\mathcal{E} = \sum_a P(a) D_{KL}(P(\underline{v} = w|a) \| \sum_h P(w|h)P(h|a)), \quad (52.8)$$

for some arbitrary choice of prior $P(a)$. For example, can choose $P(a)$ uniform.

Chapter 53

Observationally Equivalent DAGs

This chapter is based on Chapter 1 of Ref.[39] and on a blog post by Bruno Gonçalves (Ref.[11]).

A probability distribution P is **compatible with a DAG** G if P and G have the same random variables, and they can be combined to form a bnet without contradictions; i.e., one can calculate all the TPMs from P and multiply them together to obtain P again. Let

$$\mathcal{P}(G) = \{P : P \text{ is compatible with } G\} . \quad (53.1)$$

Two DAGs G and G' are observationally equivalent (OE) if $\mathcal{P}(G) = \mathcal{P}(G')$. Hence, any total probability distribution that is compatible with one of them is compatible with the other. For example, $\underline{a} \rightarrow \underline{b}$ and $\underline{a} \leftarrow \underline{b}$ are OE because

$$P(a|b)P(b) = P(a, b) = P(b|a)P(a) . \quad (53.2)$$

We'll say two bnets are OE if their DAGs are OE.

Two DAGs G and G' are **d-separation equivalent** if $DS(G) = DS(G')$. See Chapter 19 for definition of $DS(G)$.

Claim 60 *Two DAGs are OE iff their DAGs are d-separation equivalent.*

The **skeleton** of a DAG is its underlying undirected graph.

A **v-structure** in a DAG consists of two arrows converging to a node and such that their tails are not connected by a third arrow. Fig.53.1 shows in red all the v-structures of a particular DAG.

Claim 61 *Observational Equivalence Theorem (by Verma and Pearl, 1990)*

Two DAGs are OE iff they have the same skeletons and the same v-structures.

53.1 Examples

The 3 DAGs in Fig.53.2 are OE. They form an equivalence class of OE DAGs that represent the same probability distribution. This equivalence class of DAGs can be

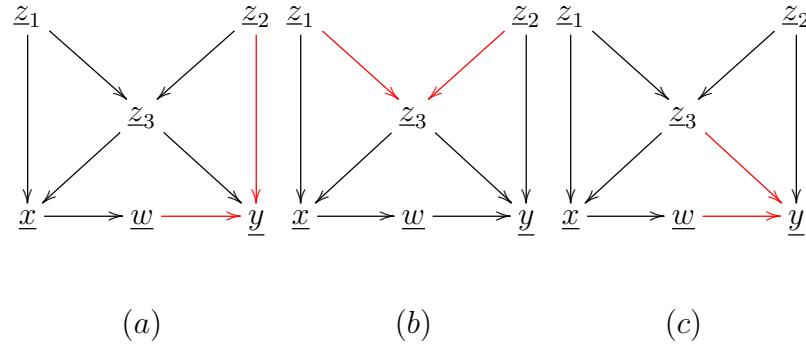


Figure 53.1: Example showing in red all v-structures of a particular DAG.

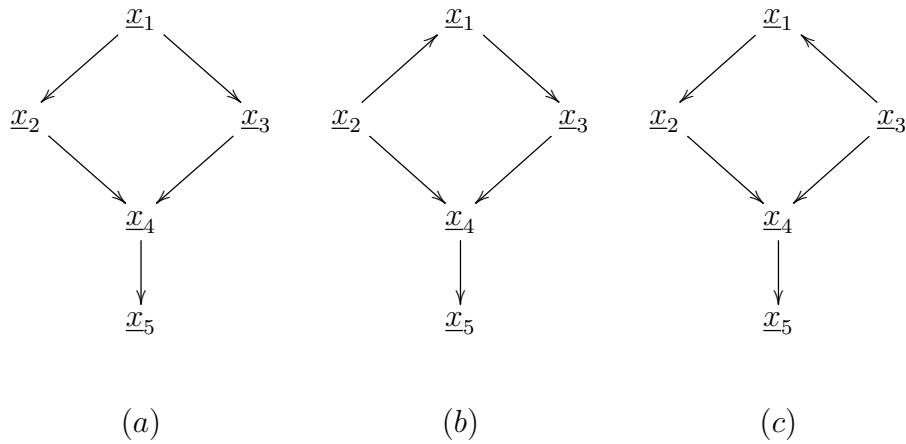


Figure 53.2: These 3 DAGs are observationally equivalent (OE).

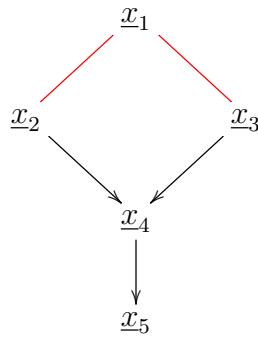


Figure 53.3: This partially directed graph represents the 3 DAGs in Fig.53.2.

represented by the partially directed graph Fig.53.3. These 3 DAGs can be proven to be OE in the following 3 ways:

1. Write the generic probability distributions represented by the 3 DAGs, and show that they are equal, as we did in Eq.(53.2). That is the low brow way of proving OE.
2. Use d-separation (see Chapter 19). Consider DAG (a) first. Rename the nodes as $\underline{\tau}_j$ with $j = 1, 2, \dots$ so that the names are in topological order (i.e., so that the parents of $\underline{\tau}_j$ have indices that are smaller than j). The node names x_j of DAG (a) are already in topological order, so we skip this step for DAG (a). Now write down its total probability distribution and notice which parents of a fully connected DAG were omitted.

$$P(x_1, x_2, x_3, x_4, x_5) = \underbrace{P(x_5|x_4)}_{x_3, x_2, x_1 \text{ omitted}} \underbrace{P(x_4|x_3, x_2)}_{x_1 \text{ omitted}} \underbrace{P(x_3|x_1)}_{x_2 \text{ omitted}} P(x_2|x_1)P(x_1) \quad (53.3)$$

The observations of which parents were omitted can be stated in d-separation lingo as the following 3 orthogonality relations:¹

$$\underline{x}_3 \perp_P \underline{x}_2 \mid \underline{x}_1 \quad (53.4a)$$

$$\underline{x}_4 \perp_P \underline{x}_1 \mid \underline{x}_2, \underline{x}_3 \quad (53.4b)$$

$$\underline{x}_5 \perp_P (\underline{x}_1, \underline{x}_2, \underline{x}_3) \& \mid \underline{x}_4. \quad (53.4c)$$

Going through the same procedure for the other 2 DAGs yields, for each of them, an equivalent set of 3 orthogonality equations.²

This is enough to conclude that the 3 DAGs of Fig.53.2 are OE.

Note that Eqs.(53.4) encompass all that there is to say about the observability of DAG (a). These 3 equations can be checked empirically to assess how well the DAG fits the data. For example, one can do OLS (ordinary least squares) regression $\underline{x}_5 \sim \underline{x}_1 + \underline{x}_2 + \underline{x}_3 + \underline{x}_4$ on the data, i.e., try to fit $x_5 = \beta_0 + \sum_{i=1}^4 \beta_i x_i$ to the data, and find that, to a good approximation, $\beta_1 = \beta_2 = \beta_3 = 0$.

3. Use the OE Theorem. All three DAGs have the same skeleton, and the same single v-structure $\underline{x}_2 \rightarrow \underline{x}_4 \leftarrow \underline{x}_3$.

¹Normally, if we had changed from the original node names to the $\underline{\tau}_j$ node names, these orthogonality relations would first be stated in terms of the $\underline{\tau}_j$ names, and we could translate them so that they were stated in terms of the original node names. But for DAG (a) there was no need to use the $\underline{\tau}_j$ names.

²The \underline{x}_j node names are no longer in topological order for DAGs (b) and (c) so for them you should go through the intermediate step of renaming the nodes $\underline{\tau}_j$, and then, after obtaining the orthogonality relations in terms of the $\underline{\tau}_j$ names, translating them back to the original \underline{x}_j names.

Chapter 54

Personalized Treatment Effects

This chapter is based on the work of Pearl et al, as reported in Refs. [56] and [28].

Recall from Chapter 56 on Potential Outcomes (PO) and Beyond, that the **Average Treatment Effect (ATE)** is defined as

$$ATE = E[\underline{y}_1 - \underline{y}_0] \quad (54.1)$$

$$= P(\underline{y}_1 = 1) - P(\underline{y}_0 = 1) \quad (54.2)$$

The **Conditional ATE (CATE)** is defined as the conditional expected value

$$ATE_z = E_{|z}[\underline{y}_1 - \underline{y}_0] \quad (54.3)$$

$$= P(\underline{y}_1 = 1|z) - P(\underline{y}_0 = 1|z) . \quad (54.4)$$

Note that

$$ATE = \sum_z P(z)ATE_z \quad (54.5)$$

Personalized Treatment Effect (PTE) theory as envisioned by Pearl is the study of bounds for “personalized” treatment effects such as

$$PNS = P(\underline{y}_1 - \underline{y}_0 = 1) \quad (54.6)$$

and

$$PNS_z = P(\underline{y}_1 - \underline{y}_0 = 1|z) . \quad (54.7)$$

They are said to be **personalized** because they are averages over a *single* ensemble (i.e., population) of individuals σ with probability $P(y_0^\sigma, y_1^\sigma)$. ATE , on the other hand, is not personalized, because it equals the difference of two of those averages.

If the conditioning z is fine grained enough to pick out a single individual σ (i.e., if $z = \sigma$), then we get

$$PNS_\sigma = \mathbb{1}(y_1^\sigma - y_0^\sigma = 1) \quad (54.8)$$

whereas

$$ATE_\sigma = \mathbb{1}(y_1^\sigma = 1) - \mathbb{1}(y_0^\sigma = 1) \quad (54.9)$$

ATE and *PNS* measure different things. *PNS* measures the probability that a *single person* will switch outcomes from 0 to 1 when he/she switches treatments from 0 to 1. *ATE* measures the difference in populations between those who survive taking the drug and those who survive without it.

One very promising field in which PTE theory can be applied is in **Personalized Causal Medicine**. For example, suppose we want to use PNS_z , where the conditioning is on the sex of the patient (i.e., $z = \text{male}, \text{female}$), to advice a female patient whether to take a cancer drug or not.

54.1 Goal, Strategy and Rationale of PTE theory

In this section, we described briefly the goal, strategy and rationale behind PTE theory.

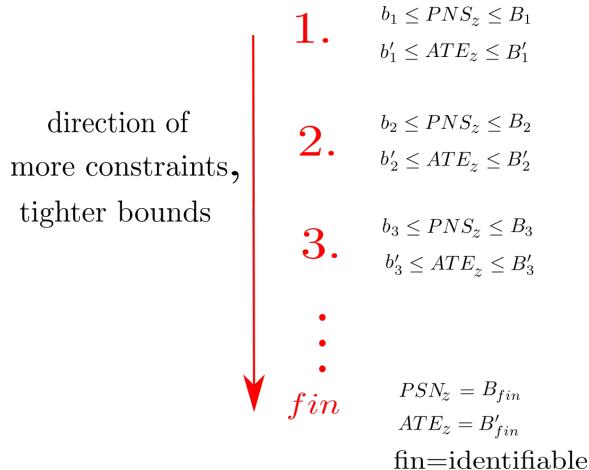


Figure 54.1: We use patient data to calculate at each step, increasingly tighter bounds b_j, b'_j, B_j, B'_j for PNS_z and ATE_z , ending in point bounds for both quantities.

The goal of PTE theory, as described by Fig.54.1, is to find increasingly tighter bounds for PTEs such as *PNS*, PNS_z , *ATE*, ATE_z , etc..

We say bounds, because it is not always possible to give a point estimate for *PNS* and other PTEs. If a point estimate for a PTE named Q is achievable, we say Q is **identifiable**. The same definition of identifiability has been used before in this book for *ATE*. It's possible that *ATE* is identifiable, but *PNS* isn't, or vice versa, for certain kinds of data; i.e., both quantities need not become identifiable simultaneous at the same step above.

The bounds given by PTE theory are as tight as possible, depending on the available data, and on what bnet model assumptions the user is willing to make. We

will consider two types of bounds: (1) Bounds for an unspecified bnet, (2) bounds for specific bnet families. Bounds for (2) will be tighter than bounds for (1).

The bounds are calculated from two types of data: **Observational Data (OD)** and **Experimental Data (ED)**.

For OD, one allows the patient to choose whether to take a drug or not ($x = 0, 1$), and then we conduct a **survey** to record his/her value for x and whether the treatment worked or not ($y = 0, 1$).

For ED, one conducts a **RCT** (Randomized Controlled Trial) instead of a survey. In the case of ED, we record, as in OD, the (x, y) for each patient, but the value of x for each patient is selected by the experimenter, at random, and, once selected, it is compulsory for the patient.

Unlike ED, OD is likely to be confounded, but it can still shed additional information that serves to tighten the bounds on PNS_z or other PTEs.

OD is usually collected first because it is easier and cheaper to collect than ED. Sometimes ED is too expensive or difficult or even impossible to collect, so only OD is available. Sometimes several ED (resp., several OD) need to be merged, before merging the merged ED with the merged OD. Pearl's PTE theory allows us to fuse together all the available data in all these types of situations.

Some purists such as the advocates of **EBM (Evidence Based Medicine)** advocate the use of only ED (i.e., only RCTs), no OD. This is reasonable in certain administrative professions to keep partisanship and chicanery out of the decision making process, but in other professions, throwing away OD would be foolish. Even in cases where an RCT is planned, a quick and cheap survey can help design a better RCT. A case in the history of medicine where OD was essential, was the case of John Snow (see Chapter 15). John Snow didn't have an RCT, but he was able to use OD to determine the driver of the cholera epidemic in London. He saved countless lives by doing so. An EBM purist would have thrown out his OD because it wasn't an RCT.

In the usual case, OD is collected first to aid in the design of an RCT, and then an RCT is conducted to collect ED. In this case, an EBM purist would throw away the OD, and only calculate an estimate of ATE. What PTE theory suggests is to calculate *both*, an estimate of ATE and bounds for PNS. Why? Because ATE and PNS measure different things. ATE utilizes only ED whereas PNS utilizes both OD and ED. Also, PNS is more personalized than ATE.

Some people object to PTE theory and CI (Causal Inference) in general on the grounds that the causal DAGs are adhoc, arbitrary, a sort of unscientific voodoo. I think it's because they fail to grasp the following 3 things:

1. DAGs are not unique. Stop thinking that you have to find the unique DAG for the situation being considered. You just have to find a DAG that is a good causal fit for the situation. If a DAG is too complicated, you can always simplify it by merging several nodes into a single more abstract one, or by summing over unwanted nodes.

2. DAGs are roughly ordered from past to present. The arrows of a DAG roughly reflect the passage of time.
3. DAGs represent a scientific hypothesis that can and should be tested with do experiments.

54.2 Bnets for PTE theory

Let $\bar{0} = 1$ and $\bar{1} = 0$.

Whenever we write $P(\underline{a} = \underline{x}, b)$, we mean $P(\underline{a} = \underline{x}, \underline{b} = b)$.

In this chapter, we will not use the notation $P(y)$ and $P(y')$ used by Pearl to discuss PTE theory. Instead, we will use $P(\underline{y} = 1)$ and $P(\underline{y} = 0)$ to denote his $P(y)$ and $P(y')$, respectively.

On the other hand, in this chapter we will change the names of variables $\underline{d}, \underline{y}, \underline{z}, \underline{y}(d)$ used in Chapter 56 on Potential Outcomes (PO) to the names favored by Pearl. Hence, we will replace $\underline{d}, \underline{y}, \underline{x}, \underline{y}(d)$ by $\underline{x}, \underline{y}, \underline{z}, \underline{y}_x$, respectively.

PTE theory considers bnets of the form Fig.54.2. The bnet considered in Rubin's PO theory is a very simple special case of this where the box labeled "multiple nodes" is absent and there is an arrow pointing from \underline{z} to \underline{x} .

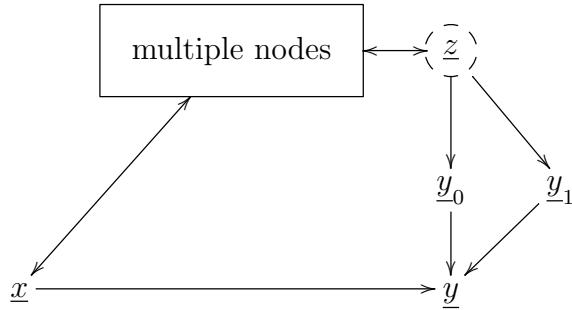


Figure 54.2: Type of Bnet considered in PTE theory. The box labeled "multiple nodes" contains various observed and hidden nodes with arrows to or from node \underline{x} and to or from node \underline{z} . \underline{z} can be a multinode. \underline{z} is shown as hidden but could be observed instead.

The TPM, printed in blue, for node \underline{y} of bnet Fig.54.2 is as follows:

$$P(\underline{y} | y_0, y_1, \underline{x}) = \delta(\underline{y}, \underline{y}_x) \quad (54.10)$$

This TPM is used frequently in PTE theory. If x, y_0, y_1 are arguments of $P()$, this TPM implies that one can swap $\underline{y}_x = y_x$ and $\underline{y} = y_x$ inside $P()$. For example,

$$P(y_0, y_1, x) = P(\underline{y}_x = y_x, y_{\bar{x}}, x) \quad (54.11)$$

$$= P(\underline{y} = y_x, y_{\bar{x}}, x) \quad (54.12)$$

According to Pearl, the defining property of y_x is that

$$P(\underline{y}_x = y) = P(y = y | \mathcal{D}\underline{x} = x) \quad (54.13)$$

Pearl likes to call Eq.(54.13) the 1st Law. The 1st Law is also a consequence of bnet Fig.54.2 and the TPM Eq.(54.10). Indeed, $\mathcal{D}\underline{x} = x$ means one should amputate all arrows entering node \underline{x} , and one should set the TPM of \underline{x} to a delta function centered at x . If that is done, then the values of \underline{y} and \underline{y}_x must be equal, because the TPM at node \underline{y} is a delta function that enforces this equality.

54.3 $ATE = PNS - AMM$

Define the **Probability of Necessity and Sufficiency**

$$PNS = P(\underline{y}_0 = 0, \underline{y}_1 = 1) \quad (54.14)$$

and the **Amonotonicity Measure** by

$$AMM = P(\underline{y}_0 = 1, \underline{y}_1 = 0) \quad (54.15)$$

Claim 62

$$PNS = P(\underline{y}_1 - \underline{y}_0 = 1) \quad (54.16)$$

$$AMM = P(\underline{y}_1 - \underline{y}_0 = -1) \quad (54.17)$$

proof:

$$\begin{aligned} \underline{y}_1 - \underline{y}_0 = 1 &\text{ iff } (\underline{y}_1 = 1 \text{ and } \underline{y}_0 = 0). \\ \underline{y}_1 - \underline{y}_0 = -1 &\text{ iff } (\underline{y}_1 = 0 \text{ and } \underline{y}_0 = 1). \end{aligned}$$

QED

Claim 63

$$ATE = PNS - AMM \quad (54.18)$$

proof:

$$ATE = E_{\sigma}[\underline{y}_1^{\sigma} - \underline{y}_0^{\sigma}] \quad (54.19)$$

$$= E[\underline{y}_1 - \underline{y}_0] \quad (54.20)$$

$$= \sum_y y [P(\underline{y}_1 = y) - P(\underline{y}_0 = y)] \quad (54.21)$$

$$= P(\underline{y}_1 = 1) - P(\underline{y}_0 = 1) \quad (54.22)$$

$$= \sum_{y_0} P(y_0, \underline{y}_1 = 1) - \sum_{y_1} P(\underline{y}_0 = 1, y_1) \quad (54.23)$$

$$= \underbrace{P(\underline{y}_0 = 0, \underline{y}_1 = 1)}_{PNS} - \underbrace{P(\underline{y}_0 = 1, \underline{y}_1 = 0)}_{AMM} \quad (54.24)$$

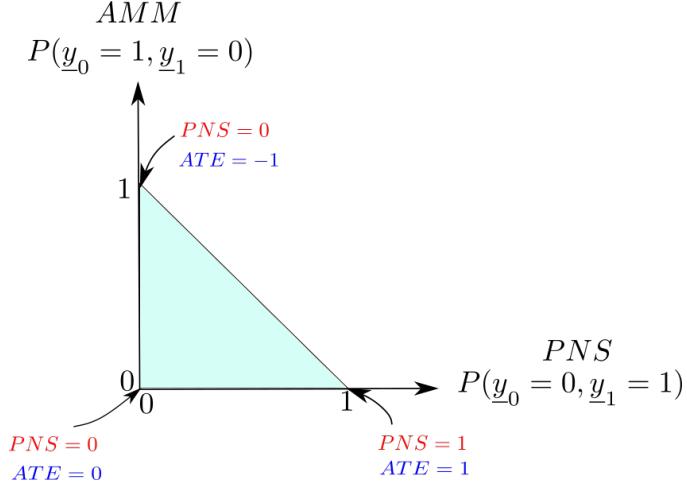


Figure 54.3: Probability simplex $\{(x, y) : x \geq 0, y \geq 0, x+y \leq 1\}$ with $x = PNS$ and $y = AMM$. Values of PNS and ATE at corners are given. Points on line $x+y=1$ can only be achieved if $P(1,1)=P(0,0)=0$.

QED

See Fig.54.3 for a visualization of the extreme values of PNS and the corresponding values of ATE .

54.4 Probabilities Relevant to PTE theory

Note¹

Let $x, y \in \{0, 1\}$ and $z \in S_z$ for some finite, not necessarily binary set S_z . Define

$$P_{x,y}^{x',y'} = P(\underline{y}_{x'} = y' | x, y) \quad (54.25)$$

observational (non-causal) probabilities

$$O_{x,y} = P(\underline{x} = x, \underline{y} = y) \quad (54.26)$$

$$O_{y|x} = P(\underline{y} = y | \underline{x} = x) \quad (54.27)$$

$$\pi_x = P(\underline{x} = x) \quad (54.28)$$

experimental (causal) probabilities

$$E_{y|x} = P(\underline{y}_x = y) \quad (54.29)$$

¹To translate this section from our notation to the notation used by Tian and Pearl in Ref.[56], replace $P(\underline{y}_1 = i, \underline{y}_0 = j, \underline{x} = k) \rightarrow p_{i,j,k} O_{1,0} \rightarrow P(x, y')$, $O_{0|1} \rightarrow P(y'|x)$, $E_{0|1} \rightarrow P(y'_x)$, etc.

Average Treatment Effect (ATE)

$$ATE = P(\underline{y}_1 = 1) - P(\underline{y}_0 = 1) \quad (54.30)$$

$$= E_{1|1} - E_{1|0} \quad (54.31)$$

$$= E_{1|1} + E_{0|0} - 1 \quad (54.32)$$

Conditional ATE (CATE)

$$ATE_z = P(\underline{y}_1 = 1|z) - P(\underline{y}_0 = 1|z) \quad (54.33)$$

$$= E_{1|1,z} + E_{0|0,z} - 1 \quad (54.34)$$

Average Causal Effect

$$ACE = P(\underline{y} = 1|\mathcal{D}\underline{x} = 1) - P(\underline{y} = 1|\mathcal{D}\underline{x} = 0) \quad (54.35)$$

Note that

$$ACE = \underbrace{P(\underline{y} = 1|\mathcal{D}\underline{x} = 1)}_{P(\underline{y}_1 = 1)} - \underbrace{P(\underline{y} = 1|\mathcal{D}\underline{x} = 0)}_{P(\underline{y}_0 = 1)} = ATE \quad (54.36)$$

Conditional ACE (CACE)

This is what is called ACE_z in Chapter 56. Note that $ACE_z = ATE_z$.

Effect of Treatment on the Treated (ETT) (i.e., ATE for the treated)

$$ETT = \underbrace{P(\underline{y}_1 = 1|\underline{x} = 1)}_{\mathcal{E}_1} - \underbrace{P(\underline{y}_0 = 1|\underline{x} = 1)}_{\mathcal{E}_0} \quad (54.37)$$

Note that

$$\mathcal{E}_1\pi_1 = P(\underline{y}_1 = 1, \underline{x} = 1) \quad (54.38)$$

$$= O_{1,1} \quad (54.39)$$

and

$$\mathcal{E}_0\pi_1 = P(\underline{y}_0 = 1, \underline{x} = 1) \quad (54.40)$$

$$= P(\underline{y}_0 = 1) - \underbrace{P(\underline{x} = 0, \underline{y}_0 = 1)}_{P(\underline{x}=0, \underline{y}=1)} \quad (54.41)$$

$$= E_{1|0} - O_{0,1} \quad (54.42)$$

so

$$ETT\pi_1 = \sum_x O_{x,1} - E_{1|0} \quad (54.43)$$

Probability of Necessity (PN)²

$$PN = P_{1,1}^{0,0} \quad (54.44)$$

Probability of Sufficiency (PS)³

$$PS = P_{0,0}^{1,1} \quad (54.45)$$

Probability of Necessity and Sufficiency (PNS)

$$PNS = P(\underline{y}_0 = 0, \underline{y}_1 = 1) \quad (54.46)$$

Henceforth, we will use $PNS3$ to denote the trio

$$PNS3 = (PNS, PN, PS) . \quad (54.47)$$

Amontonicity Measure (AMM)

$$AMM = P(\underline{y}_0 = 1, \underline{y}_1 = 0) \quad (54.48)$$

Risk ratio or relative risk (RR)

$$RR = \frac{O_{1|1}}{O_{1|0}} \quad (54.49)$$

Excess Risk Ratio (ERR)

$$ERR = \frac{O_{1|1} - O_{1|0}}{O_{1|1}} = 1 - \frac{1}{RR} \quad (54.50)$$

Corrected ERR (CERR)

$$CERR = ERR + \frac{O_{1|0} - E_{1|0}}{O_{1,1}} \quad (54.51)$$

You might be wondering how $P(\underline{y}_x = y|x = 0)$ and $P(\underline{y}_x = y|\underline{x} = 1)$ for $x, y \in \{0, 1\}^2$ are related to $O_{y|x}$ and $E_{y|x}$. The following claim shows how.

Claim 64

$$P(\underline{y}_x = y|x = x) = O_{y|x} \quad (54.52a)$$

$$P(\underline{y}_x = y|x = \bar{x}) = \frac{E_{y|x} - O_{y|x}\pi_x}{\pi_{\bar{x}}} \quad (54.52b)$$

²I like to call PN the Probability of Nullifying, because it goes from 11 to 00

³I like to call PS the Probability of Surging, because it goes from 00 to 11

proof: Before we begin the proof, note that summing both sides of Eqs.(54.52) gives $1 = 1$, so these 2 equations pass that test.

$$P(\underline{y}_x = y | \underline{x} = x) = P(\underline{y} = y | \underline{x} = x) = O_{y|x} \quad (54.53)$$

$$P(\underline{y}_x = y | \underline{x} = \bar{x}) = \frac{P(\underline{y}_x = y, \underline{x} = \bar{x})}{\pi_{\bar{x}}} \quad (54.54)$$

$$= \frac{P(\underline{y}_x = y) - P(\underline{y}_x = y, x = x)}{\pi_{\bar{x}}} \quad (54.55)$$

$$= \frac{P(\underline{y}_x = y) - P(\underline{y}_x = y | \underline{x} = x)\pi_x}{\pi_{\bar{x}}} \quad (54.56)$$

$$= \frac{E_{y|x} - O_{y|x}\pi_x}{\pi_{\bar{x}}} \quad (54.57)$$

QED

Claim 65

$$P(y_0, y_1 | x) = P(y_{\bar{x}} | x, \underline{y} = y_x)P(\underline{y} = y_x | x) \quad (54.58)$$

$$= \begin{array}{c} y_{\bar{x}} \\ \nearrow \\ x \longrightarrow \underline{y} = y_x \end{array} \quad (54.59)$$

proof:

$$P(y_0, y_1 | x) = \frac{P(y_0, y_1, x)}{P(x)} \quad (54.60)$$

$$= \frac{P(y_{\bar{x}}, x, \underline{y} = y_x)}{P(x)} \quad (54.61)$$

$$= \frac{P(y_{\bar{x}} | x, \underline{y} = y_x)P(x, \underline{y} = y_x)}{P(x)} \quad (54.62)$$

$$= P(y_{\bar{x}} | x, \underline{y} = y_x)P(\underline{y} = y_x | x) \quad (54.63)$$

QED

Claim 66

$$\underbrace{PNS}_{P(\underline{y}_0=0, \underline{y}_1=1)} = PN * O_{1,1} + PS * O_{0,0} \quad (54.64)$$

Hence, if we know any two of (PN, PS, PNS) , we can calculate the third.

proof:

$$P(y_0, y_1) = \sum_x P(y_0, y_1|x)P(x) \quad (54.65)$$

$$= \sum_x P(y_{\bar{x}}|x, \underline{y} = y_x)P(x, \underline{y} = y_x) \quad (\text{see Claim 65.}) \quad (54.66)$$

$$= \begin{cases} P(y_1|\underline{x} = 0, \underline{y} = y_0)P(\underline{x} = 0, \underline{y} = y_0) \\ + P(y_0|\underline{x} = 1, \underline{y} = y_1)P(\underline{x} = 1, \underline{y} = y_1) \end{cases} \quad (54.67)$$

Thus,

$$P(\underline{y}_0 = 0, \underline{y}_1 = 1) = \begin{cases} P(\underline{y}_1 = 1|\underline{x} = 0, \underline{y} = 0)P(\underline{x} = 0, \underline{y} = 0) \\ + P(\underline{y}_0 = 0|\underline{x} = 1, \underline{y} = 1)P(\underline{x} = 1, \underline{y} = 1) \end{cases} \quad (54.68)$$

$$= PS * O_{0,0} + PN * O_{1,1} \quad (54.69)$$

QED

Note that PNS refers to both \underline{y}_0 and \underline{y}_1 , PN refers only to \underline{y}_0 and PS refers only to \underline{y}_1 . Thus, PN and PS serve to separate the pair of variables $(\underline{y}_0, \underline{y}_1)$ and to isolate them individually. Claim 66 is a quantitative expression of that separation.

Pearl likes to say that PNS belongs to Rung 3 because it's a probability that involves both \underline{y}_0 and \underline{y}_1 , and one of those two must be a counterfactual (an event that never occurred). On the other hand, PN, PS, ATE , etc., are defined in terms of probabilities that involve either \underline{y}_0 or \underline{y}_1 but not both. Probabilities that involve only one of them, can be expressed with the do operator, so they belong to Rung 2. Conditional probabilities like $P(y|x)$ that involve neither \underline{y}_0 nor \underline{y}_1 belong to Rung 1.⁴

54.5 Symmetry

Define \sim to be an operator that swaps zeros and ones in $P(\underline{y}_0 = y, \underline{y}_1 = y', x)$. Hence

$$[P(\underline{y}_0 = y, \underline{y}_1 = y', \underline{x} = x)]^\sim = P(\underline{y}_1 = \bar{y}, \underline{y}_0 = \bar{y}', \underline{x} = \bar{x}) \quad (54.70)$$

$$(P_{x,y}^{x',y'})^\sim = P_{\bar{x},\bar{y}}^{\bar{x}',\bar{y}'} \quad (54.71)$$

$$(O_{x,y})^\sim = O_{\bar{x},\bar{y}} \quad (54.72)$$

$$(O_{y|x})^\sim = O_{\bar{y}|\bar{x}} \quad (54.73)$$

⁴Probabilities such as $P(\underline{y}_0 = 1, \underline{y} = 0, \underline{x} = 1) = P(\underline{y}_0 = 1, \underline{y}_1 = 0, \underline{x} = 1)$ are considered Rung 3.

$$(\pi_x)^\sim = \pi_{\bar{x}} \quad (54.74)$$

$$(E_{y|x})^\sim = E_{\bar{y}|\bar{x}} \quad (54.75)$$

$$(PN)^\sim = PS, \quad (PS)^\sim = PN \quad (54.76)$$

$$(PNS)^\sim = PNS \quad (54.77)$$

$$(RR)^\sim = \frac{O_{0|0}}{O_{0|1}} \quad (54.78)$$

Recall

$$ERR = \frac{O_{1|1} - O_{1|0}}{O_{1|1}} = 1 - \frac{1}{RR} \quad (54.79)$$

Therefore, define

$$(ERR)^\sim = \frac{O_{0|0} - O_{0|1}}{O_{0|0}} = 1 - \frac{1}{(RR)^\sim} \quad (54.80)$$

Note that

$$O_{1|1} - O_{1|0} = O_{1|1} + O_{0|0} - 1 \quad (54.81)$$

$$= O_{0|0} - O_{0|1} \quad (54.82)$$

$$= (O_{1|1} - O_{1|0})^\sim \quad (54.83)$$

54.6 Linear Programming Problem

Probability Simplex

$$\mathcal{S} = \left\{ P(y_0, y_1, x) : \begin{array}{l} P(y_0, y_1, x) \geq 0 \\ y_0, y_1, x \in \{0, 1\} \\ \sum_{y_0=0}^1 \sum_{y_1=0}^1 \sum_{x=0}^1 P(y_0, y_1, x) = 1 \end{array} \right\} \quad (54.84)$$

Note that \mathcal{S} has 7 degrees of freedom (dofs).

1. observational (non-causal) constraints on \mathcal{S} (3 constraints)

$$O_{x,y} = P(x, y) = \sum_{y'} P(\underline{y}_x = y, \underline{y}_{\bar{x}} = y', x) \quad \text{for } (x, y) \in \{(0, 1), (1, 0), (1, 1)\} \quad (54.85)$$

2. experimental (causal) constraints on \mathcal{S} (2 constraints)⁵

$$E_{1|x} = P(\underline{y}_x = 1) = \sum_{x'} \sum_{y'} P(\underline{y}_x = 1, \underline{y}_{\bar{x}} = y', x = x') \quad \text{for } x \in \{0, 1\} \quad (54.86)$$

\mathcal{S} has 7 dofs but the 3 observational constraints reduce the number of dofs to 4. The 5 observational and experimental constraints reduce the number of dofs to 2. \mathcal{S} is embedded in \mathbb{R}^8 and then the 6 constraints (unit probability, 2 observational and 3 experimental) reduce it to the interior of a 6 or less sided figure in \mathbb{R}^2 .

Henceforth, we will refer to the observational and experimental constraints together as the **minimal constraints**.

This is half of a linear programming problem. Recall that a **linear programming problem** can be stated as finding the column vector ξ that minimizes a cost $\mathcal{C} = c^T \xi$ subject to $A\xi = b$ and $\xi \geq 0$. Here we have no cost function or minimization, but we have $A\xi = b$ and $\xi \geq 0$ where $\xi = [P(y_0, y_1, x)]_{\forall y_0, y_1, x}$. Also $b = [1, O_{0,1}, O_{1,0}, O_{1,1}, E_{1|0}, E_{1|1}]^T$, and A = a matrix of zeros and ones.

Note that

$$PNS = P(\underline{y}_0 = 0, \underline{y}_1 = 1) \quad (54.87)$$

$$= \sum_x P(\underline{y}_0 = 0, \underline{y}_1 = 1, x) \quad (54.88)$$

$$PN = P(\underline{y}_0 = 0 | \underline{x} = 1, \underline{y} = 1) \quad (54.89)$$

$$= \frac{P(\underline{y}_0 = 0, \underline{y}_1 = 1, \underline{x} = 1)}{O_{1,1}} \quad (54.90)$$

$$PS = P(\underline{y}_1 = 1 | \underline{x} = 0, \underline{y} = 0) \quad (54.91)$$

$$= \frac{P(\underline{y}_0 = 0, \underline{y}_1 = 1, \underline{x} = 0)}{O_{0,0}} \quad (54.92)$$

54.7 Special constraints

- **Exogeneity (a.k.a., no-confounding)** holds for simplex \mathcal{S} if $\underline{y}_x \perp \underline{x}$ for $x \in \{0, 1\}$. Hence

$$\underbrace{P(\underline{y}_x = 1)}_{E_{1|x}} = P(\underline{y}_x = 1 | x) = \underbrace{P(\underline{y} = 1 | x)}_{O_{1|x}} \quad \text{for } x \in \{0, 1\} \quad (54.93)$$

⁵ $E_{0|x}$ follows from $E_{0|x} = 1 - E_{1|x}$.

Exogeneity gives **2 constraints**.

Note that exogeneity is the same thing as identifiability of $P(\underline{y}_x = \underline{y}) = P(\underline{y} = \underline{y} | \mathcal{D}\underline{x} = \underline{x})$. But identifiability (i.e., do-identifiability) is a more general concept. One can speak of the identifiability of $P(y_x | z)$ or of $P(y_0, y_1, a)$, etc. In general, any expression with do operators is do-identifiable if it can be expressed as an expression without do-operators.

- **Strong Exogeneity** holds for simplex \mathcal{S} if $(\underline{y}_0, \underline{y}_1)_{joint} \perp \underline{x}$. Hence

$$P(y_0, y_1 | x) = P(y_0, y_1) \quad (54.94)$$

Strong exogeneity gives **3 constraints**. Strong exogeneity implies exogeneity but not the converse.⁶

- **Monotonicity**⁷ holds for simplex \mathcal{S} if

$$AMM = P(\underline{y}_0 = 1, \underline{y}_1 = 0) = 0 \quad (54.95)$$

. Equivalently,

$$\sum_x P(\underline{y}_0 = 1, \underline{y}_1 = 0, x) = 0 \quad (54.96)$$

which is true iff

$$P(\underline{y}_0 = 1, \underline{y}_1 = 0, x) = 0 \quad \text{for } x \in \{0, 1\} \quad (54.97)$$

Monotonicity gives **2 constraints**.

Note that when $AMM = 0$, $PNS = ATE$.

Claim 67 *Monotonicity and exogeneity together imply strong exogeneity.*

proof:

This proof is presented here for completeness. Later on, we will give a much simpler proof of this result. I advise the reader to skip this proof on first reading of this chapter.

⁶In this book, we use both $(\underline{a}, \underline{b})_\& \perp \underline{x}$ and $(\underline{a}, \underline{b})_{joint} \perp \underline{x}$. When we write $(\underline{a}, \underline{b})_\& \perp \underline{x}$, we mean that $\underline{a} \perp \underline{x}$ and $\underline{b} \perp \underline{x}$, or, equivalently, $P(a|x) = P(a)$ and $P(b|x) = P(b)$. When we write $(\underline{a}, \underline{b})_{joint} \perp \underline{x}$ or simply $(\underline{a}, \underline{b}) \perp \underline{x}$, we mean $P(a, b|x) = P(a, b)$. Summing $P(a, b|x) = P(a, b)$ over a gives $P(b|x) = P(b)$, and summing it over b gives $P(a|x) = P(a)$. Hence we see that $(\underline{a}, \underline{b})_{joint} \perp \underline{x}$ implies $(\underline{a}, \underline{b})_\& \perp \underline{x}$ but not the converse. If \underline{a} and \underline{b} are independent at fixed \underline{x} so that $P(a, b|x) = P(a|x)P(b|x)$ then $(\underline{a}, \underline{b})_{joint} \perp \underline{x}$ iff $(\underline{a}, \underline{b})_\& \perp \underline{x}$

⁷This property is called monotonicity because it's equivalent to the statement that $y_0^\sigma \leq y_1^\sigma$ for all individuals σ in the population. Indeed, $y_0^\sigma \leq y_1^\sigma$ includes the 3 cases $(y_0^\sigma, y_1^\sigma) = (0, 0), (1, 1), (0, 1)$, but excludes the only other case, namely $(1, 0)$.

Let $a \in \{0, 1\}$.

$$P(\underline{y}_a = \bar{a}, x) = \sum_{a'} P(\underline{y}_a = \bar{a}, \underline{y}_{\bar{a}} = a', x) \quad (54.98)$$

$$= P(\underline{y}_a = \bar{a}, \underline{y}_{\bar{a}} = \bar{a}, x) \quad (\text{by monotonicity}) \quad (54.99)$$

Thus,

$$P(\underline{y}_a = \bar{a} | \underline{x} = a) = P(\underline{y}_a = \bar{a}, \underline{y}_{\bar{a}} = \bar{a} | \underline{x} = a) \quad (54.100)$$

and

$$P(\underline{y}_a = \bar{a}) = P(\underline{y}_a = \bar{a}, \underline{y}_{\bar{a}} = \bar{a}) \quad (54.101)$$

By exogeneity, the left hand side of Eq.(54.100) and the left hand side of Eq.(54.101) are equal, so the right hand sides of those equations must be equal too.

$$P(\underline{y}_a = \bar{a}, \underline{y}_{\bar{a}} = \bar{a} | \underline{x} = a) = P(\underline{y}_a = \bar{a}, \underline{y}_{\bar{a}} = \bar{a}) \quad (54.102)$$

This immediately implies⁸

$$P(\underline{y}_a = \bar{a}, \underline{y}_{\bar{a}} = \bar{a} | x) = P(\underline{y}_a = \bar{a}, \underline{y}_{\bar{a}} = \bar{a}) \quad (54.103)$$

for $x \in \{0, 1\}$. This gives for $a = 0$

$$P(\underline{y}_0 = 1, \underline{y}_1 = 1 | x) = P(\underline{y}_0 = 1, \underline{y}_1 = 1) \quad (54.104)$$

and for $a = 1$

$$\underbrace{P(\underline{y}_1 = 0, \underline{y}_0 = 0 | x)}_{P(\underline{y}_0 = 0, \underline{y}_1 = 0 | x)} = \underbrace{P(\underline{y}_1 = 0, \underline{y}_0 = 0)}_{P(\underline{y}_0 = 0, \underline{y}_1 = 0)} \quad (54.105)$$

Monotonicity itself gives

$$P(\underline{y}_0 = 1, \underline{y}_1 = 0 | x) = 0 = P(\underline{y}_0 = 1, \underline{y}_1 = 0) \quad (54.106)$$

The remaining strong exogeneity constraint, given by

$$P(\underline{y}_0 = 0, \underline{y}_1 = 1 | x) = P(\underline{y}_0 = 0, \underline{y}_1 = 1), \quad (54.107)$$

follows because

$$\sum_{y_0, y_1} P(y_0, y_1 | x) = \sum_{y_0, y_1} P(y_0, y_1) = 1 \quad (54.108)$$

QED

⁸If $x \in \{0, 1\}$ and $P(a | \underline{x} = 0) = P(a)$, then $P(a) - P(a, \underline{x} = 0) = P(a)[1 - P(\underline{x} = 0)]$, so $P(a, \underline{x} = 1) = P(a)P(\underline{x} = 1)$. Hence, $P(a | \underline{x} = 1) = P(a)$.

54.8 Matrix representation of probabilities

Suppose $\epsilon_x(y_0, y_1)$ for $x, y_0, y_1 \in \{0, 1\}$ are eight vectors in a vector space V . $\epsilon_x(y_0, y_1)$ will only be used at position (y_0, y_1) of a 2×2 matrix, where the positions are defined as follows:

$$\begin{array}{ccc} & y_1 & \\ \uparrow & & \\ 1 & (0, 1) & (1, 1) \\ | & & \\ 0 & (0, 0) & (1, 0) \\ | & & \\ 0 & \longrightarrow & 1 \longrightarrow y_0 \end{array} \quad (54.109)$$

When $\epsilon_x(y_0, y_1)$ is used inside a 2×2 matrix, we will not write the argument (y_0, y_1) , leaving it implicit, unless confusion may arise. Thus, for example, we will represent

$$\mathcal{P} \begin{bmatrix} \epsilon_0(0, 1) & 0 \\ 0 & 0 \end{bmatrix} \text{ by } \mathcal{P} \begin{bmatrix} \epsilon_0 & 0 \\ 0 & 0 \end{bmatrix}.$$

For $x \in \{0, 1\}$, let

$$P_{\underline{y}_0, \underline{y}_1, \underline{x}}(0, 0, x) = \mathcal{P} \begin{bmatrix} 0 & 0 \\ \epsilon_x & 0 \end{bmatrix} \quad (54.110a)$$

$$P_{\underline{y}_0, \underline{y}_1, \underline{x}}(0, 1, x) = \mathcal{P} \begin{bmatrix} \epsilon_x & 0 \\ 0 & 0 \end{bmatrix} \quad (54.110b)$$

$$P_{\underline{y}_0, \underline{y}_1, \underline{x}}(1, 0, x) = \mathcal{P} \begin{bmatrix} 0 & 0 \\ 0 & \epsilon_x \end{bmatrix} \quad (54.110c)$$

$$P_{\underline{y}_0, \underline{y}_1, \underline{x}}(1, 1, x) = \mathcal{P} \begin{bmatrix} 0 & \epsilon_x \\ 0 & 0 \end{bmatrix} \quad (54.110d)$$

Define a map $\mathcal{P}[\quad] : V \rightarrow \mathbb{R}$ that we will call the **matrix representation of probabilities (MRP)** (mnemonic, Mr. P). We will assume that the map $\mathcal{P}[\quad]$ is linear. Hence, for instance,

$$\mathcal{P} \begin{bmatrix} 4\epsilon_0 + 3\epsilon_1 & -5\epsilon_0 \\ 0 & 0 \end{bmatrix} = 4\mathcal{P} \begin{bmatrix} \epsilon_0 & 0 \\ 0 & 0 \end{bmatrix} + 3\mathcal{P} \begin{bmatrix} \epsilon_1 & 0 \\ 0 & 0 \end{bmatrix} - 5\mathcal{P} \begin{bmatrix} 0 & \epsilon_0 \\ 0 & 0 \end{bmatrix} \quad (54.111)$$

Henceforth, we will use the abbreviation

$$\epsilon_+ = \epsilon_0 + \epsilon_1 \quad (54.112)$$

Note that

$$\mathcal{P} \begin{bmatrix} \epsilon_+ & \epsilon_+ \\ \epsilon_+ & \epsilon_+ \end{bmatrix} = 1 \quad (54.113)$$

$$\pi_x = \mathcal{P} \begin{bmatrix} \epsilon_x & \epsilon_x \\ \epsilon_x & \epsilon_x \end{bmatrix} \quad (54.114)$$

$$E_{0|0} = \mathcal{P} \begin{bmatrix} \epsilon_+ & 0 \\ \epsilon_+ & 0 \end{bmatrix}, \quad O_{0,0} = \mathcal{P} \begin{bmatrix} \epsilon_0 & 0 \\ \epsilon_0 & 0 \end{bmatrix} \quad (54.115)$$

$$E_{0|1} = \mathcal{P} \begin{bmatrix} 0 & 0 \\ \epsilon_+ & \epsilon_+ \end{bmatrix}, \quad O_{1,0} = \mathcal{P} \begin{bmatrix} 0 & 0 \\ \epsilon_1 & \epsilon_1 \end{bmatrix} \quad (54.116)$$

$$E_{1|0} = \mathcal{P} \begin{bmatrix} 0 & \epsilon_+ \\ 0 & \epsilon_+ \end{bmatrix}, \quad O_{0,1} = \mathcal{P} \begin{bmatrix} 0 & \epsilon_0 \\ 0 & \epsilon_0 \end{bmatrix} \quad (54.117)$$

$$E_{1|1} = \mathcal{P} \begin{bmatrix} \epsilon_+ & \epsilon_+ \\ 0 & 0 \end{bmatrix}, \quad O_{1,1} = \mathcal{P} \begin{bmatrix} \epsilon_1 & \epsilon_1 \\ 0 & 0 \end{bmatrix} \quad (54.118)$$

$$P(\underline{y} = 0) = O_{0,0} + O_{1,0} = \mathcal{P} \begin{bmatrix} \epsilon_0 & 0 \\ \epsilon_+ & \epsilon_1 \end{bmatrix} \quad (54.119)$$

$$P(\underline{y} = 1) = O_{0,1} + O_{1,1} = \mathcal{P} \begin{bmatrix} \epsilon_1 & \epsilon_+ \\ 0 & \epsilon_0 \end{bmatrix} \quad (54.120)$$

$$PN * O_{1,1} = \mathcal{P} \begin{bmatrix} \epsilon_0 & 0 \\ 0 & 0 \end{bmatrix} \quad (54.121)$$

$$PS * O_{0,0} = \mathcal{P} \begin{bmatrix} \epsilon_1 & 0 \\ 0 & 0 \end{bmatrix} \quad (54.122)$$

$$PNS = \mathcal{P} \begin{bmatrix} \epsilon_+ & 0 \\ 0 & 0 \end{bmatrix} \quad (54.123)$$

$$AMM = \mathcal{P} \begin{bmatrix} 0 & 0 \\ 0 & \epsilon_+ \end{bmatrix} \quad (54.124)$$

$$ATE = PNS - AMM = \mathcal{P} \begin{bmatrix} \epsilon_+ & 0 \\ 0 & -\epsilon_+ \end{bmatrix} \quad (54.125)$$

When $AMM = 0$,

$$P(\underline{y} = 1)|_{AMM=0} = \mathcal{P} \begin{bmatrix} \epsilon_1 & \epsilon_+ \\ 0 & 0 \end{bmatrix} \quad (54.126)$$

The special constraints of exogeneity, strong exogeneity and monotonicity have a very simple in the MRP.

Suppose that $\pi_0, \pi_1 \geq 1$ and $\pi_0 + \pi_1 = 1$. Note that

$$\left. \begin{array}{l} \epsilon_+ = \frac{\epsilon_0}{\pi_0} \\ \text{or} \\ \epsilon_+ = \frac{\epsilon_1}{\pi_1} \end{array} \right\} \implies \epsilon_+ = \frac{\epsilon_0}{\pi_0} = \frac{\epsilon_1}{\pi_1} \quad (54.127)$$

Below, we will abbreviate

$$\hat{\epsilon}_x = \frac{\epsilon_x}{\pi_x} \quad (54.128)$$

for $x \in \{0, 1\}$.

Claim 68

(a) *Exogeneity holds iff (“4 sides can change color”)*

$$\mathcal{P} \begin{bmatrix} \hat{\epsilon}_0 & 0 \\ \hat{\epsilon}_0 & 0 \end{bmatrix} = \mathcal{P} \begin{bmatrix} \hat{\epsilon}_1 & 0 \\ \hat{\epsilon}_1 & 0 \end{bmatrix} = \mathcal{P} \begin{bmatrix} \epsilon_+ & 0 \\ \epsilon_+ & 0 \end{bmatrix} \quad (54.129a)$$

$$\mathcal{P} \begin{bmatrix} \hat{\epsilon}_0 & \hat{\epsilon}_0 \\ 0 & 0 \end{bmatrix} = \mathcal{P} \begin{bmatrix} \hat{\epsilon}_1 & \hat{\epsilon}_1 \\ 0 & 0 \end{bmatrix} = \mathcal{P} \begin{bmatrix} \epsilon_+ & \epsilon_+ \\ 0 & 0 \end{bmatrix} \quad (54.129b)$$

$$\mathcal{P} \begin{bmatrix} 0 & \hat{\epsilon}_0 \\ 0 & \hat{\epsilon}_0 \end{bmatrix} = \mathcal{P} \begin{bmatrix} 0 & \hat{\epsilon}_1 \\ 0 & \hat{\epsilon}_1 \end{bmatrix} = \mathcal{P} \begin{bmatrix} 0 & \epsilon_+ \\ 0 & \epsilon_+ \end{bmatrix} \quad (54.129c)$$

$$\mathcal{P} \begin{bmatrix} 0 & 0 \\ \hat{\epsilon}_0 & \hat{\epsilon}_0 \end{bmatrix} = \mathcal{P} \begin{bmatrix} 0 & 0 \\ \hat{\epsilon}_1 & \hat{\epsilon}_1 \end{bmatrix} \mathcal{P} \begin{bmatrix} 0 & 0 \\ \epsilon_+ & \epsilon_+ \end{bmatrix} \quad (54.129d)$$

(b) *Monotonicity holds iff*

$$\mathcal{P} \begin{bmatrix} 0 & 0 \\ 0 & \epsilon_x \end{bmatrix} = 0 \quad (54.130)$$

for $x \in \{0, 1\}$.

(c) *Strong exogeneity holds iff (“4 corners can change color”)*

$$\mathcal{P} \begin{bmatrix} \hat{\epsilon}_0 & 0 \\ 0 & 0 \end{bmatrix} = \mathcal{P} \begin{bmatrix} \hat{\epsilon}_1 & 0 \\ 0 & 0 \end{bmatrix} = \mathcal{P} \begin{bmatrix} \epsilon_+ & 0 \\ 0 & 0 \end{bmatrix} \quad (54.131a)$$

$$\mathcal{P} \begin{bmatrix} 0 & \hat{\epsilon}_0 \\ 0 & 0 \end{bmatrix} = \mathcal{P} \begin{bmatrix} 0 & \hat{\epsilon}_1 \\ 0 & 0 \end{bmatrix} = \mathcal{P} \begin{bmatrix} 0 & \epsilon_+ \\ 0 & 0 \end{bmatrix} \quad (54.131b)$$

$$\mathcal{P} \begin{bmatrix} 0 & 0 \\ 0 & \hat{\epsilon}_0 \end{bmatrix} = \mathcal{P} \begin{bmatrix} 0 & 0 \\ 0 & \hat{\epsilon}_1 \end{bmatrix} = \mathcal{P} \begin{bmatrix} 0 & 0 \\ 0 & \epsilon_+ \end{bmatrix} \quad (54.131c)$$

$$\mathcal{P} \begin{bmatrix} 0 & 0 \\ \hat{\epsilon}_0 & 0 \end{bmatrix} = \mathcal{P} \begin{bmatrix} 0 & 0 \\ \hat{\epsilon}_1 & 0 \end{bmatrix} = \mathcal{P} \begin{bmatrix} 0 & 0 \\ \epsilon_+ & 0 \end{bmatrix} \quad (54.131d)$$

proof:

(a) This follows from the definitions of $E_{y|x}$ and $O_{x,y}$ for $x, y \in \{0, 1\}$, as stated above in the MRP.

(b) This follows from the identity

$$P_{\underline{y}_0, \underline{y}_1, \underline{x}}(1, 0|x) = \mathcal{P} \begin{bmatrix} 0 & 0 \\ 0 & \frac{\epsilon_x}{\pi_x} \end{bmatrix} \quad (54.132)$$

for $x \in \{0, 1\}$.

(c) This follows from definitions of $P_{\underline{y}_0, \underline{y}_1, \underline{x}}(y_0, y_1, x)$ for $y_0, y_1, x \in \{0, 1\}$, as stated above in the MRP.

QED

Note that it is obvious from Claim 68 that strong exogeneity implies exogeneity.

Claim 68 also makes it easy peasy to prove that exogeneity and monotonicity imply strong exogeneity. Indeed, here is a much simpler proof than the one we presented earlier.

Claim 69 *Exogeneity and monotonicity imply strong exogeneity.*

proof: Let's call $E(a), E(b), E(c), E(d)$ the four Eqs.(54.129) that define Exogeneity, and $SE(a), SE(b), SE(c), SE(d)$ the four Eqs.(54.131) that define Strong Exogeneity.

- Monotonicity implies the lower right entry is zero, so $SE(c)$ is true.
- Setting to zero the lower right entry in $E(c)$ and $E(d)$ implies $SE(b)$ and $SE(d)$.
- Subtracting $SE(d)$ from $E(a)$ gives $SE(a)$.

QED

54.9 Bounds on Exp. Probs. imposed by Obs. Probs.

Claim 70 *In general,*

$$O_{x,1} \leq E_{1|x} \leq 1 - O_{x,0} \quad \text{for } x \in \{0, 1\} . \quad (54.133)$$

In other words,

$$O_{1,1} \leq E_{1|1} \leq 1 - O_{1,0} \quad (54.134)$$

$$O_{0,1} \leq E_{1|0} \leq 1 - O_{0,0} \quad (54.135)$$

With monotonicity, we get the tighter bounds

$$\overbrace{O_{1,1} + O_{0,1}}^{P(\underline{y}=1)} \leq E_{1|1} \leq 1 - O_{1,0} \quad (54.136)$$

$$O_{0,1} \leq E_{1|0} \leq \overbrace{1 - O_{0,0} - O_{1,0}}^{P(\underline{y}=1)} \quad (54.137)$$

proof:

$$O_{1,1} \leq E_{1|1} \leq \underbrace{1 - O_{1,0}}_{O_{1,1} + O_{0,0} + O_{0,1}} \quad (54.138)$$

has the MRP

$$\mathcal{P} \begin{bmatrix} \epsilon_1 & \epsilon_1 \\ 0 & 0 \end{bmatrix} \leq \mathcal{P} \begin{bmatrix} \epsilon_+ & \epsilon_+ \\ 0 & 0 \end{bmatrix} \leq \mathcal{P} \begin{bmatrix} \epsilon_1 & \epsilon_1 \\ 0 & 0 \end{bmatrix} + \mathcal{P} \begin{bmatrix} \epsilon_0 & 0 \\ \epsilon_0 & 0 \end{bmatrix} + \mathcal{P} \begin{bmatrix} 0 & \epsilon_0 \\ 0 & \epsilon_0 \end{bmatrix} \quad (54.139)$$

which is obviously true.

$$O_{0,1} \leq E_{1|0} \leq \underbrace{1 - O_{0,0}}_{O_{1,1} + O_{0,1} + O_{1,0}} \quad (54.140)$$

has the MRP

$$\mathcal{P} \begin{bmatrix} 0 & \epsilon_0 \\ 0 & \epsilon_0 \end{bmatrix} \leq \mathcal{P} \begin{bmatrix} 0 & \epsilon_+ \\ 0 & \epsilon_+ \end{bmatrix} \leq \mathcal{P} \begin{bmatrix} \epsilon_1 & \epsilon_1 \\ 0 & 0 \end{bmatrix} + \mathcal{P} \begin{bmatrix} 0 & \epsilon_0 \\ 0 & \epsilon_0 \end{bmatrix} + \mathcal{P} \begin{bmatrix} 0 & 0 \\ \epsilon_1 & \epsilon_1 \end{bmatrix} \quad (54.141)$$

which is obviously true.

$$P(\underline{y} = 1)|_{AMM=0} \leq E_{1|1} \quad (54.142)$$

has the MRP

$$\mathcal{P} \begin{bmatrix} \epsilon_1 & \epsilon_+ \\ 0 & 0 \end{bmatrix} \leq \mathcal{P} \begin{bmatrix} \epsilon_+ & \epsilon_+ \\ 0 & 0 \end{bmatrix} \quad (54.143)$$

which is obviously true.

$$E_{1|0}|_{AMM=0} \leq P(\underline{y} = 1)|_{AMM=0} \quad (54.144)$$

has the MRP

$$\mathcal{P} \begin{bmatrix} 0 & \epsilon_+ \\ 0 & 0 \end{bmatrix} \leq \mathcal{P} \begin{bmatrix} \epsilon_1 & \epsilon_+ \\ 0 & 0 \end{bmatrix} \quad (54.145)$$

which is obviously true.

QED

54.10 Bounds on PNS_3 for unspecified bnet

Claim 71 If $P(a, b)$ is a probability distribution, then

$$\max\{0, P(a) + P(b) - 1\} \leq P(a, b) \leq \min\{P(a), P(b)\} \quad (54.146)$$

proof: $P(a|b) \leq 1$ and $P(b|a) \leq 1$ implies $P(a, b) \leq P(b)$ and $P(a, b) \leq P(a)$. Hence, $P(a, b) \leq \min\{P(a), P(b)\}$.

$$P(a) + P(b) - 1 = \sum_{a', b'} P(a', b') \underbrace{[\delta(a, a') + \delta(b, b') - 1]}_T \quad (54.147)$$

$$\leq P(a, b) \quad (T \text{ biggest when } a = a' \text{ and } b = b') \quad (54.148)$$

QED

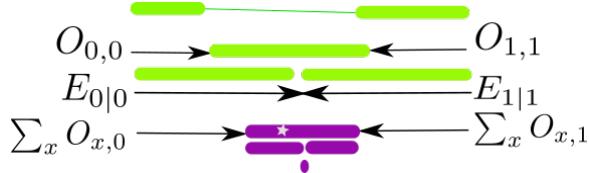


Figure 54.4: Minimal bounds for PNS . PNS is larger than maximum of the purple segments and smaller than the minimum of the green ones. The purple segment of almost zero length represents zero. The green segment interrupted by a thin green line represents $O_{0,0} + O_{1,1}$. Note that $\sum_x O_{x,0} + \sum_x O_{x,1} = 1$. When monotonicity holds, PNS equals $E_{1|1} + E_{0|0} - 1$ which is the purple segment marked with a star.

Claim 72 Minimal bounds (see Fig.54.4)

If the minimal constraints hold for simplex \mathcal{S} , then

$$\max \left\{ \begin{array}{l} 0 \\ E_{1|1} + E_{0|0} - 1 \\ E_{0|0} - \sum_x O_{x,0} \\ E_{1|1} - \sum_x O_{x,1} \end{array} \right\} \leq PNS \leq \min \left\{ \begin{array}{l} E_{1|1} \\ E_{0|0} \\ O_{1,1} + O_{0,0} \\ E_{1|1} + E_{0|0} - O_{1,1} - O_{0,0} \end{array} \right\} \quad (54.149)$$

$$\max \left\{ \frac{0}{E_{0|0} - \sum_x O_{x,0}} \right\} \leq PN \leq \min \left\{ \frac{1}{E_{0|0} - O_{0,0}} \right\} \quad (54.150)$$

$$\max \left\{ \frac{0}{E_{1|1} - \sum_x O_{x,1}} \right\} \leq PS \leq \min \left\{ \frac{1}{E_{1|1} - O_{1,1}} \right\} \quad (54.151)$$

proof: These bounds were copied directly from Ref.[56], except the notation was changed. In certain cases, we have slightly rewritten the bounds from Ref.[56] to exhibit more explicitly their symmetry under swaps of zeros and ones. For example, instead of using $(E_{1|0}, E_{1|1})$ as parameters, we use $(E_{1|1}, E_{0|0})$.

These bounds are obviously true in the MRP. Indeed, in the MRP, the following holds. For PNS , we have

$$\begin{bmatrix} 0 \\ E_{1|1} + E_{0|0} - 1 \\ E_{0|0} - \sum_x O_{x,0} \\ E_{1|1} - \sum_x O_{x,1} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathcal{P} \begin{bmatrix} \epsilon_+ & 0 \\ 0 & -\epsilon_+ \end{bmatrix} \\ \mathcal{P} \begin{bmatrix} \epsilon_1 & 0 \\ 0 & -\epsilon_1 \end{bmatrix} \\ \mathcal{P} \begin{bmatrix} \epsilon_0 & 0 \\ 0 & -\epsilon_0 \end{bmatrix} \end{bmatrix} \quad (54.152)$$

$$PNS = \mathcal{P} \begin{bmatrix} \epsilon_+ & 0 \\ 0 & 0 \end{bmatrix} \quad (54.153)$$

$$\begin{bmatrix} E_{1|1} \\ E_{0|0} \\ O_{1,1} + O_{0,0} \\ E_{1|1} + E_{0|0} - O_{1,1} - O_{0,0} \end{bmatrix} = \begin{bmatrix} \mathcal{P} \begin{bmatrix} \epsilon_+ & 0 \\ \epsilon_+ & 0 \end{bmatrix} \\ \mathcal{P} \begin{bmatrix} \epsilon_+ & \epsilon_+ \\ 0 & 0 \end{bmatrix} \\ \mathcal{P} \begin{bmatrix} \epsilon_+ & \epsilon_1 \\ \epsilon_0 & 0 \end{bmatrix} \\ \mathcal{P} \begin{bmatrix} \epsilon_+ & \epsilon_0 \\ \epsilon_1 & 0 \end{bmatrix} \end{bmatrix} \quad (54.154)$$

For PN , we have

$$\begin{bmatrix} 0 \\ E_{1|1} - \sum_x O_{x,1} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathcal{P} \begin{bmatrix} \epsilon_0 & 0 \\ 0 & -\epsilon_0 \end{bmatrix} \end{bmatrix} \quad (54.155)$$

$$PN * O_{1,1} = \mathcal{P} \begin{bmatrix} \epsilon_0 & 0 \\ 0 & 0 \end{bmatrix} \quad (54.156)$$

$$\begin{bmatrix} 1 \\ E_{1|1} - O_{1,1} \end{bmatrix} = \begin{bmatrix} 1 \\ \mathcal{P} \begin{bmatrix} \epsilon_0 & \epsilon_0 \\ 0 & 0 \end{bmatrix} \end{bmatrix} \quad (54.157)$$

For PS , we have

$$\begin{bmatrix} 0 \\ E_{0|0} - \sum_x O_{x,0} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathcal{P} \begin{bmatrix} \epsilon_1 & 0 \\ 0 & -\epsilon_1 \end{bmatrix} \end{bmatrix} \quad (54.158)$$

$$PS * O_{0,0} = \mathcal{P} \begin{bmatrix} \epsilon_1 & 0 \\ 0 & 0 \end{bmatrix} \quad (54.159)$$

$$\begin{bmatrix} 1 \\ E_{0|0} - O_{0,0} \end{bmatrix} = \begin{bmatrix} 1 \\ \mathcal{P} \begin{bmatrix} \epsilon_1 & \epsilon_1 \\ 0 & 0 \end{bmatrix} \end{bmatrix} \quad (54.160)$$

The first two lines of the bound for PNS are a simple consequence of Claim 71. Indeed, Claim 71 implies

$$\max \left\{ \frac{0}{P(y_0) + P(y_1) - 1} \right\} \leq P(y_0, y_1) \leq \min \left\{ \frac{P(y_0)}{P(y_1)} \right\} \quad (54.161)$$

When $y_0 = 0, y_1 = 1$, we get

$$\max \left\{ \frac{0}{E_{0|0} + E_{1|1} - 1} \right\} \leq PNS \leq \min \left\{ \frac{E_{0|0}}{E_{1|1}} \right\} \quad (54.162)$$

QED

In Claim 72, note that

- $\sum_x O_{x,y} = P(\underline{y} = y)$ for $y = 0, 1$.
- Let

$$E'_{0|0} = E_{0|0} - \sum_x O_{x,0} = \mathcal{P} \begin{bmatrix} \epsilon_1 & 0 \\ 0 & -\epsilon_1 \end{bmatrix} \quad (54.163)$$

$$E'_{1|1} = E_{1|1} - \sum_x O_{x,1} = \mathcal{P} \begin{bmatrix} \epsilon_0 & 0 \\ 0 & -\epsilon_0 \end{bmatrix} \quad (54.164)$$

$$ATE = E_{1|1} + E_{0|0} - 1 = E'_{1|1} + E'_{0|0} = \mathcal{P} \begin{bmatrix} \epsilon_+ & 0 \\ 0 & -\epsilon_+ \end{bmatrix} \quad (54.165)$$

- Claim 72 applies if we have both observational data (OD) and experimental data (ED). If we only have OD (resp., ED), ignore all bounds that involve an E (resp., involve an O) probability.

Hence, with only OD⁹

$$0 \leq PNS \leq O_{1,1} + O_{0,0} \quad (54.166)$$

⁹The bounds Eq.(54.166) assume exogeneity does not hold. For the case when exogeneity does hold, stronger bounds will be given later on in the chapter.

and $PN, PS \in [0, 1]$, whereas with only ED,

$$\max \left\{ \frac{0}{E_{1|1} + E_{0|0} - 1} \right\} \leq PNS \leq \min \left\{ \frac{E_{1|1}}{E_{0|0}} \right\} \quad (54.167)$$

and $PN, PS \in [0, 1]$.

- At first blush, there seem to be 3 possible cases to consider: (1) Only OD. (2) Both OD and ED. (3) Only ED. Actually, Case (1) plus the assumption of exogeneity is the same as Case (3). Indeed, recall that exogeneity means no confounding, and ED (i.e., an RCT) also has no confounding. So we don't have to consider Case (3) if we consider Case (1) without and with exogeneity. Exogeneity is built into case (2), so case (2) without exogeneity is meaningless.

Claim 73 *If minimal and exogeneity constraints hold for simplex \mathcal{S} , then*

$$\max \left\{ \frac{0}{O_{1|1} + O_{0|0} - 1}, \frac{O_{0|0} - \sum_x O_{x,0}}{O_{1|1} - \sum_x O_{x,1}} \right\} \leq PNS \leq \min \left\{ \frac{O_{1|1}}{O_{0|0}}, \frac{O_{1,1} + O_{0,0}}{O_{1|1} + O_{0|0} - O_{1,1} - O_{0,0}} \right\} \quad (54.168)$$

$$\max \left\{ \frac{0}{ERR} \right\} \leq PN \leq \min \left\{ \frac{1}{\frac{O_{0|0}}{O_{1|1}}} \right\} \quad (54.169)$$

$$\max \left\{ \frac{0}{(ERR)^\sim} \right\} \leq PS \leq \min \left\{ \frac{1}{\frac{O_{1|1}}{O_{0|0}}} \right\} \quad (54.170)$$

proof: Just replace $E_{y|x}$ by $O_{y|x}$ in the minimal bounds given in Claim 72.

The canceled terms do not improve the bounds and can be dropped. We show this next using MRP.

In the MRP, the minimal bounds for PNS are

$$\begin{bmatrix} 0 \\ E_{1|1} + E_{0|0} - 1 \\ E_{0|0} - \sum_x O_{x,0} \\ E_{1|1} - \sum_x O_{x,1} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathcal{P} \begin{bmatrix} \epsilon_+ & 0 \\ 0 & -\epsilon_+ \end{bmatrix} \\ \mathcal{P} \begin{bmatrix} \epsilon_1 & 0 \\ 0 & -\epsilon_1 \end{bmatrix} \\ \mathcal{P} \begin{bmatrix} \epsilon_0 & 0 \\ 0 & -\epsilon_0 \end{bmatrix} \end{bmatrix} \quad (54.171)$$

$$PNS = \mathcal{P} \begin{bmatrix} \epsilon_+ & 0 \\ 0 & 0 \end{bmatrix} \quad (54.172)$$

$$\begin{bmatrix} E_{1|1} \\ E_{0|0} \\ O_{1,1} + O_{0,0} \\ E_{1|1} + E_{0|0} - O_{1,1} - O_{0,0} \end{bmatrix} = \begin{bmatrix} \mathcal{P} \begin{bmatrix} \epsilon_+ & 0 \\ \epsilon_+ & 0 \end{bmatrix} \\ \mathcal{P} \begin{bmatrix} \epsilon_+ & \epsilon_+ \\ 0 & 0 \end{bmatrix} \\ \mathcal{P} \begin{bmatrix} \epsilon_+ & \epsilon_1 \\ \epsilon_0 & 0 \end{bmatrix} \\ \mathcal{P} \begin{bmatrix} \epsilon_+ & \epsilon_0 \\ \epsilon_1 & 0 \end{bmatrix} \end{bmatrix} \quad (54.173)$$

If exogeneity holds, we can replace all E 's by O 's on the left hand sides. We can also use $\epsilon_+ = \hat{\epsilon}_0 = \hat{\epsilon}_1$ on the right hand sides to conclude that

$$O_{1|1} + O_{0|0} - 1 = \mathcal{P} \begin{bmatrix} \epsilon_+ & \epsilon_+ - \epsilon_+ \\ 0 & -\epsilon_+ \end{bmatrix} = \mathcal{P} \begin{bmatrix} \hat{\epsilon}_x & \hat{\epsilon}_x - \hat{\epsilon}_x \\ 0 & -\hat{\epsilon}_x \end{bmatrix} = \frac{1}{\pi_x} \mathcal{P} \begin{bmatrix} \epsilon_x & 0 \\ 0 & -\epsilon_x \end{bmatrix} \quad (54.174)$$

for both $x \in \{0, 1\}$. Note that $\frac{1}{\pi_x} \mathcal{P} \begin{bmatrix} \epsilon_x & 0 \\ 0 & -\epsilon_x \end{bmatrix}$ will always be greater or equal to $\mathcal{P} \begin{bmatrix} \epsilon_x & 0 \\ 0 & -\epsilon_x \end{bmatrix}$ when $\mathcal{P} \begin{bmatrix} \epsilon_x & 0 \\ 0 & -\epsilon_x \end{bmatrix} \geq 0$ (if it's negative, the 0 bound takes precedence). Hence, the two canceled lower bound terms can be dropped. A similar argument shows that the two canceled upper bound terms can be dropped too.

QED

Claim 74 *If the minimal and strong exogeneity constraints hold for simplex \mathcal{S} , then the inequalities for exogeneity Claim 73 hold. In addition,*

$$PN = \frac{PNS}{O_{1|1}} \quad (54.175)$$

and

$$PS = \frac{PNS}{O_{0|0}} \quad (54.176)$$

proof:

$$PN * O_{1|1} = P(\underline{y}_0 = 0, \underline{y} = 1 | \underline{x} = 1) \quad (54.177)$$

$$= P(\underline{y}_0 = 0, \underline{y}_1 = 1) \quad (54.178)$$

$$= PNS \quad (54.179)$$

$$PS * O_{0|0} = P(\underline{y}_1 = 1, \underline{y} = 0 | \underline{x} = 0) \quad (54.180)$$

$$= P(\underline{y}_0 = 0, \underline{y}_1 = 1) \quad (54.181)$$

$$= PNS \quad (54.182)$$

QED

Claim 75 *If the minimal and monotonicity constraints hold for simplex \mathcal{S} , then*

$$PNS = E_{1|1} + E_{0|0} - 1 \quad (54.183)$$

$$PN = \frac{E_{0|0} - \sum_x O_{x,0}}{O_{1,1}} = \frac{\sum_x O_{x,1} - E_{1|0}}{O_{1,1}} = CERR \quad (54.184)$$

$$PS = \frac{E_{1|1} - \sum_x O_{x,1}}{O_{0,0}} = \frac{\sum_x O_{x,0} - E_{0|1}}{O_{0,0}} = (CERR)^\sim \quad (54.185)$$

proof: These results can be easily proven using the MRP.

Note that

$$\frac{\sum_x O_{x,1} - E_{1|0}}{O_{1,1}} = \frac{O_{1|1}\pi_1 + O_{1|0}(1 - \pi_1) - E_{1|0}}{O_{1|1}\pi_1} \quad (54.186)$$

$$= 1 - \frac{O_{1|0}}{O_{1|1}} + \frac{O_{1|0} - E_{1|0}}{O_{1,1}} \quad (54.187)$$

$$= CERR \quad (54.188)$$

QED

Claim 76 *If the minimal, exogeneity and monotonicity constraints hold for simplex \mathcal{S} , then PNS, PN, PS are identifiable, and*

$$PNS = O_{1|1} + O_{0|0} - 1 \quad (54.189)$$

$$PN = \frac{O_{0|0} - \sum_x O_{x,0}}{O_{1,1}} = \frac{\sum_x O_{x,1} - O_{1|0}}{O_{1,1}} = ERR \quad (54.190)$$

$$PS = \frac{O_{1|1} - \sum_x O_{x,1}}{O_{0,0}} = \frac{\sum_x O_{x,0} - O_{0|1}}{O_{0,0}} = (ERR)^\sim \quad (54.191)$$

proof: Set $E_{y|x} = O_{y|x}$ in Claim 75.

QED

54.11 Bounds on PNS_3 for specific bnet families

Define

$$E_{*,z} = E_{0|0,z} + E_{1|1,z} = ATE_z + 1 \quad (54.192)$$

$$O_{*,z} = O_{0,0|z} + O_{1,1|z} \quad (54.193)$$

Claim 77 If $\underline{z} \cap de(\underline{x}) = \emptyset$, then

$$\max \left\{ \frac{0}{E_{0|0,z} - \sum_x O_{x,0|z}}, \frac{E_{*,z} - 1}{E_{1|1,z} - \sum_x O_{x,1|z}} \right\} \leq PNS_z \leq \min \left\{ \frac{E_{1|1,z}}{E_{0|0,z}}, \frac{O_{*,z}}{E_{*,z} - O_{*,z}} \right\} \quad (54.194)$$

$$\max \left\{ \frac{0}{\frac{E_{0|0,z} - \sum_x O_{x,0|z}}{O_{1,1|z}}} \right\} \leq PN_z \leq \min \left\{ \frac{1}{\frac{E_{0|0,z} - O_{0,0|z}}{O_{1,1|z}}} \right\} \quad (54.195)$$

$$\max \left\{ \frac{0}{\frac{E_{1|1,z} - \sum_x O_{x,1|z}}{O_{0,0|z}}} \right\} \leq PS_z \leq \min \left\{ \frac{1}{\frac{E_{1|1,z} - O_{1,1|z}}{O_{0,0|z}}} \right\} \quad (54.196)$$

proof: These are just the minimal bounds where everything is conditioned on z .

QED

In the minimal bounds, node \underline{z} is not mentioned because the minimal bounds don't assume that \underline{z} exists. Once we assume that node \underline{z} exists, then we can use it, as Claim 77 does.

Claim 78 If \underline{z} satisfies the backdoor adjustment criterion relative $(\underline{x}, \underline{y})$, then

$$\max \left\{ \frac{0}{\frac{O_{*,z} - 1}{\cancel{O_{0|0,z} - \sum_x O_{x,0|z}}}}, \frac{\cancel{O_{0|0,z} - \sum_x O_{x,0|z}}}{\cancel{O_{1|1,z} - \sum_x O_{x,1|z}}} \right\} \leq PNS_z \leq \min \left\{ \frac{O_{1|1,z}}{\cancel{O_{0|0,z}}}, \frac{\cancel{O_{*,z}}}{\cancel{O_{*,z} - O_{*,z}}} \right\} \quad (54.197)$$

$$\max \left\{ \frac{0}{\frac{O_{0|0,z} - \sum_x O_{x,0|z}}{O_{1,1|z}}} \right\} \leq PN_z \leq \min \left\{ \frac{1}{\frac{O_{0|0,z} - O_{0,0|z}}{O_{1,1|z}}} \right\} \quad (54.198)$$

$$\max \left\{ \frac{0}{\frac{O_{1|1,z} - \sum_x O_{x,1|z}}{O_{0,0|z}}} \right\} \leq PS_z \leq \min \left\{ \frac{1}{\frac{O_{1|1,z} - O_{1,1|z}}{O_{0,0|z}}} \right\} \quad (54.199)$$

proof: Satisfaction of the backdoor criterion implies that

$$\underbrace{P(y|x, z)}_{O_{y|x,z}} = \underbrace{P(\underline{y}_x = y|z)}_{E_{y|x,z}} \quad (54.200)$$

Replace $E_{y|x,z}$ by $O_{y|x,z}$ in Claim 77. These are just the minimal bounds, when exogeneity holds, that were considered previously. As before, the canceled terms do not affect the bounds so they can be dropped.

QED

So far we have considered 2 families of bnets: (1) $\underline{z} \cap de(\underline{x}) = \emptyset$ and (2) \underline{z} satisfies backdoor adjustment criterion. Actually, (2) is a subset of (1). Next, we will consider bnets in which \underline{z} is a mediator relative to $(\underline{x}, \underline{y})$. A mediator \underline{z} is impinged by an arrow $\underline{x} \rightarrow \underline{z}$, so it definitely does not satisfy (1). Bnets that do not satisfy (1) also fall outside the purview of Potential Outcomes (PO) theory.

We say \underline{z} is a **partial mediator** relative to $(\underline{x}, \underline{y})$ if $\underline{y}_x \perp (\underline{x}, \underline{z}_{\bar{x}}) \& | \underline{z}_x$ for $x \in \{0, 1\}$. See Fig.54.5 for an example. In that figure, the TPMs, printed in blue, for nodes \underline{z} and \underline{y} , are as follows.

$$P(z|x, z_0, z_1) = \delta(z, z_x) \quad (54.201)$$

$$P(y|x, y_0, y_1) = \delta(y, y_x) \quad (54.202)$$

If \underline{z} is a partial mediator relative to $(\underline{x}, \underline{y})$ and there is no direct path from \underline{x} to \underline{y} , we say \underline{x} is a **pure mediator** relative to $(\underline{x}, \underline{y})$.

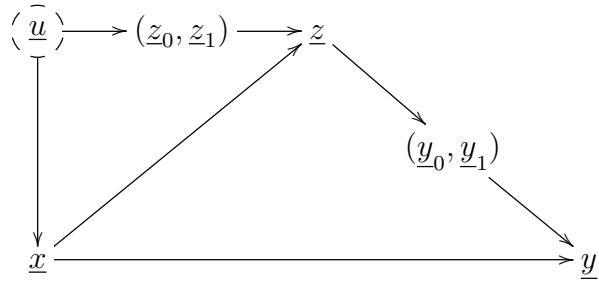


Figure 54.5: Example of a bnet in which \underline{z} is a partial mediator relative to $(\underline{x}, \underline{y})$. If $\underline{x} = 1$, then conditioning on \underline{z}_1 blocks the paths from \underline{y}_1 to \underline{x} and from \underline{y}_1 to \underline{z}_0 . If \underline{z} is a partial mediator relative to $(\underline{x}, \underline{y})$, and, in addition, there is no arrow $\underline{x} \rightarrow \underline{y}$, we say \underline{z} is a pure mediator relative to $(\underline{x}, \underline{y})$.

Claim 79 (*Bounds if \underline{z} is partial mediator*)

If \underline{z} is a partial mediator relative to $(\underline{x}, \underline{y})$, and $\underline{x}, \underline{y}, \underline{z} \in \{0, 1\}$, then

$$\max \left\{ \begin{array}{l} 0 \\ \frac{E_{*|*}-1}{E_{0|0}-\sum_x O_{x,0}} \\ \frac{E_{1|1}-\sum_x O_{x,1}}{E_{*|*}-O_{*,*}} \end{array} \right\} \leq PNS \leq \min \left\{ \begin{array}{l} E_{1|1} \\ E_{0|0} \\ O_{*,*} \\ new \end{array} \right\} \quad (54.203)$$

where

$$new = \sum_{z_0, z_1} \min \left\{ \begin{array}{l} O_{1|1, z_1} \\ O_{0|0, z_0} \end{array} \right\} \min \left\{ \begin{array}{l} P(z_1) \\ P(z_0) \end{array} \right\} \quad (54.204)$$

proof:

$$PNS = P(\underline{y}_1 = 1, \underline{y}_0 = 0) \quad (54.205)$$

$$= \sum_{z_0, z_1} P(\underline{y}_1 = 1, \underline{y}_0 = 0 | z_1, z_0) P(z_1, z_0) \quad (54.206)$$

$$\leq \sum_{z_0, z_1} \min \left\{ \begin{array}{l} P(\underline{y}_1 = 1 | z_1, z_0) \\ P(\underline{y}_0 = 0 | z_1, z_0) \end{array} \right\} \min \left\{ \begin{array}{l} P(z_1) \\ P(z_0) \end{array} \right\} \quad (54.207)$$

$$= \sum_{z_0, z_1} \min \left\{ \begin{array}{l} P(\underline{y} = 1 | z_1) \\ P(\underline{y} = 0 | z_0) \end{array} \right\} \min \left\{ \begin{array}{l} P(z_1) \\ P(z_0) \end{array} \right\} \text{(because } \underline{y}_x \perp \underline{z}_{\bar{x}} | \underline{z}_x) \quad (54.208)$$

$$= \sum_{z_0, z_1} \min \left\{ \begin{array}{l} P(\underline{y} = 1 | \underline{x} = 1, z_1) \\ P(\underline{y} = 0 | \underline{x} = 0, z_0) \end{array} \right\} \min \left\{ \begin{array}{l} P(z_1) \\ P(z_0) \end{array} \right\} \text{(because } \underline{y}_x \perp \underline{x} | \underline{z}_x) \quad (54.209)$$

$$= \sum_{z_0, z_1} \min \left\{ \begin{array}{l} P(\underline{y} = 1 | \underline{x} = 1, \underline{z} = z_1) \\ P(\underline{y} = 0 | \underline{x} = 0, \underline{z} = z_0) \end{array} \right\} \min \left\{ \begin{array}{l} P(z_1) \\ P(z_0) \end{array} \right\} \quad (54.210)$$

(because $\underline{z} = \underline{z}_x$ at fixed x)

$$= \sum_{z_0, z_1} \min \left\{ \begin{array}{l} O_{1|1, z_1} \\ O_{0|0, z_0} \end{array} \right\} \min \left\{ \begin{array}{l} P(z_1) \\ P(z_0) \end{array} \right\} \quad (54.211)$$

QED

Claim 80 (*Bounds if \underline{z} is a pure mediator*)

If \underline{z} is a pure mediator relative to $(\underline{x}, \underline{y})$, and $\underline{x}, \underline{y}, \underline{z} \in \{0, 1\}$, then Claim 79 for a partial mediator is still valid, except that now “new” equals

$$new = \sum_{z_0, z_1} \mathbb{1}(z_0 \neq z_1) \min \left\{ \begin{array}{l} O_{1|\underline{x}=z_1} \\ O_{0|\underline{x}=z_0} \end{array} \right\} \min \left\{ \begin{array}{l} P(\underline{z} = z_1 | \underline{x} = 1) \\ P(\underline{z} = z_0 | \underline{x} = 0) \end{array} \right\} \quad (54.212)$$

proof:

$$PNS = P(\underline{y}_1 = 1, \underline{y}_0 = 0) \quad (54.213)$$

$$= \sum_{z_0, z_1} \mathbb{1}(z_0 \neq z_1) P(\underline{y}_{z_1} = 1, \underline{y}_{z_0} = 0) P(z_1, z_0) \quad (54.214)$$

(because $(\underline{y}_0, \underline{y}_1) \perp (z_0, z_1)$ and $\underline{y}_0 \neq \underline{y}_1$)

$$\leq \sum_{z_0, z_1} \mathbb{1}(z_0 \neq z_1) \min \left\{ \begin{array}{l} P(\underline{y}_{z_1} = 1) \\ P(\underline{y}_{z_0} = 0) \end{array} \right\} \min \left\{ \begin{array}{l} P(z_1) \\ P(z_0) \end{array} \right\} \quad (54.215)$$

$$= \sum_{z_0, z_1} \mathbb{1}(z_0 \neq z_1) \min \left\{ \begin{array}{l} P(\underline{y} = 1 | \underline{x} = z_1) \\ P(\underline{y} = 0 | \underline{x} = z_0) \end{array} \right\} \min \left\{ \begin{array}{l} P(z_1 | \underline{x} = 1) \\ P(z_0 | \underline{x} = 0) \end{array} \right\} \quad (54.216)$$

(because $\underline{y} = \underline{y}_x$ at fixed \underline{x} and $(z_0, z_1) \perp \underline{x}$)

$$= \sum_{z_0, z_1} \mathbb{1}(z_0 \neq z_1) \min \left\{ \begin{array}{l} O_{1|\underline{x}=z_1} \\ O_{0|\underline{x}=z_0} \end{array} \right\} \min \left\{ \begin{array}{l} P(z = z_1 | \underline{x} = 1) \\ P(z = z_0 | \underline{x} = 0) \end{array} \right\} \quad (54.217)$$

QED

54.12 Numerical Examples

I've written an open source Python program (See Ref[60]) that calculates the bounds given in this chapter. The program is called "JudeasRx", in honor of Judea Pearl. Fig.54.6 is an example of its interface with data entered by Boris Sobolev from a real life case. JudeasRx considers $z = g = \text{gender} \in \{m, f\}$.

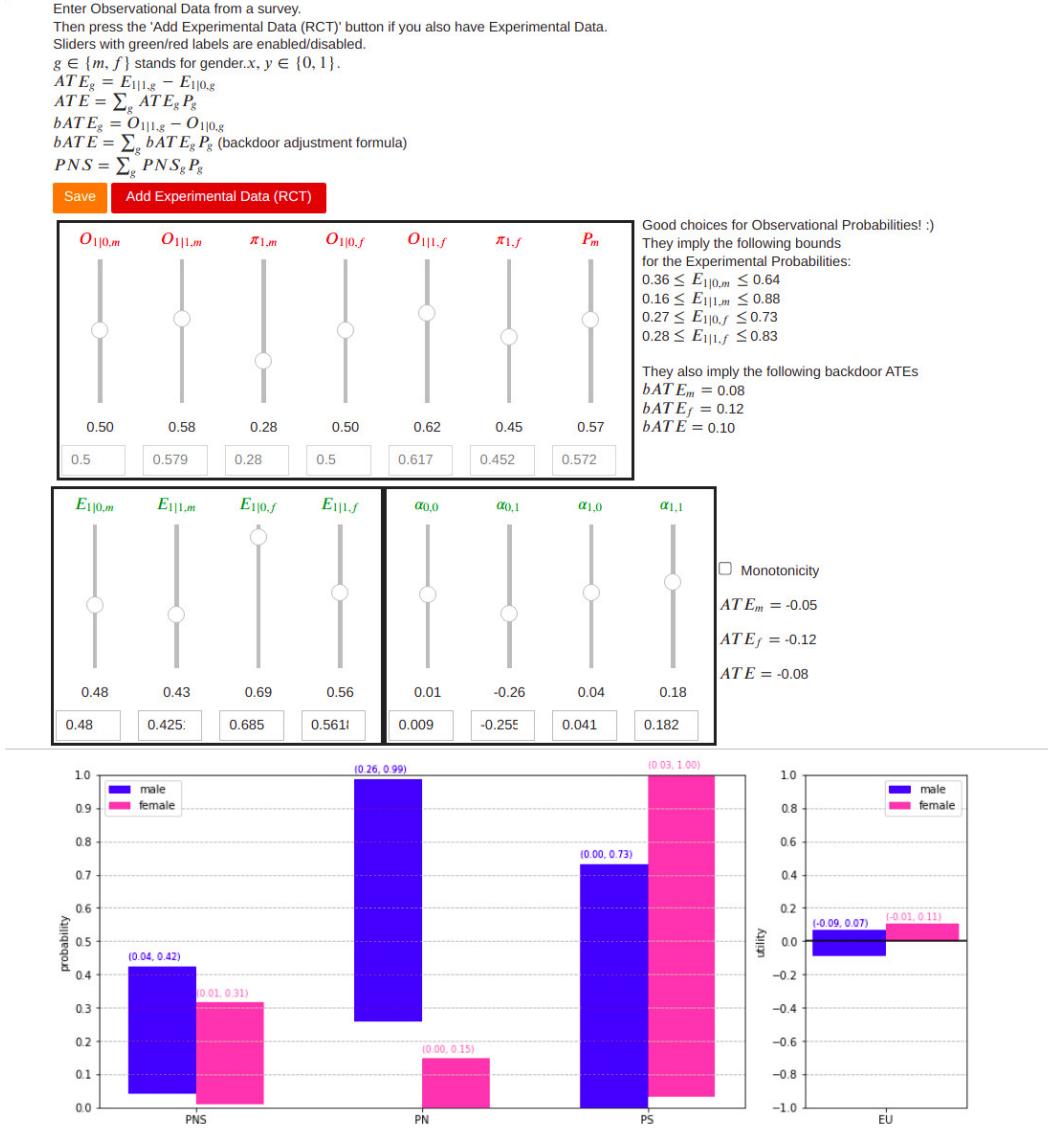


Figure 54.6: Interface for the Python program JudeasRx, with data entered by Boris Sobolev.

Chapter 55

Personalized Expected Utility

This chapter is based on Ref.[22].

This chapter assumes that the reader has already read Chapter 54 on Personalized Treatment Effects. Whereas Chapter 54 is concerned with finding bounds for PNS_z , this chapter will find bounds for EU_z , which is called the Personalized Expected Utility (PEU).

For $y_0, y_1 \in \{0, 1\}$, let us denote the **conditional joint experimental (causal, counterfactual) distribution** by:

$$P_{y_0, y_1 | z} = P(\underline{y}_0 = y_0, \underline{y}_1 = y_1 | z). \quad (55.1)$$

Suppose we are given a **Utility function**¹

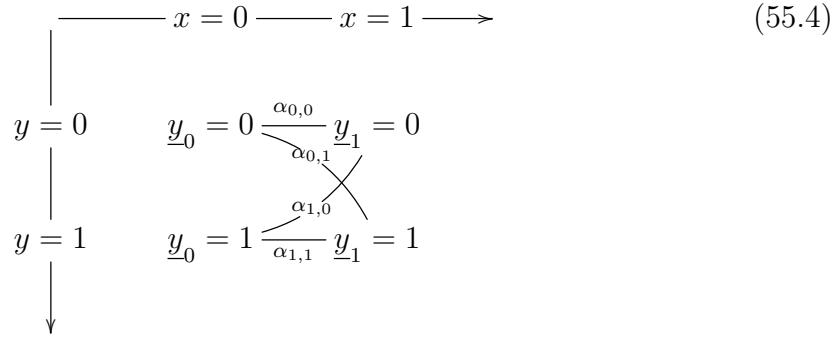
$$\alpha_{y_0, y_1} : \{0, 1\}^2 \rightarrow \mathbb{R} \quad (55.2)$$

Let²

$$\begin{aligned} \beta &= \alpha_{0,1} & (y_0 = 0, y_1 = 1) && \text{compliers} \\ \gamma &= \alpha_{1,1} & (y_0 = 1, y_1 = 1) && \text{always takers} \\ \theta &= \alpha_{0,0} & (y_0 = 0, y_1 = 0) && \text{never takers} \\ \delta &= \alpha_{1,0} & (y_0 = 1, y_1 = 0) && \text{defiers} \end{aligned} \quad (55.3)$$

¹Ref.[22] refers to the utility function as the benefit function.

²The notation $\beta, \gamma, \theta, \delta$ won't be used again in this chapter. We mention it here so the reader can translate to and from our equations and the equations in Ref.[22] where this $\beta, \gamma, \theta, \delta$ notation is used.



Define the **personalized expected utility (PEU)** by

$$EU = E[\alpha_{\underline{y}_0, \underline{y}_1}] = \alpha_{i,j} P_{i,j} \quad (55.5)$$

and the **conditional PEU** by

$$EU_z = E_{|z}[\alpha_{\underline{y}_0, \underline{y}_1}] = \alpha_{i,j} P_{i,j|z} \quad (55.6)$$

Note that

$$EU = \sum_z P(z) EU_z. \quad (55.7)$$

Above, we are using the Einstein summation convention (repeated indices are to be summed over) for the indices $i, j \in \{0, 1\}$.

Compare the definition of EU_z with that of two other causal effects used in his book:

$$PNS_z = P_{0,1||z} = P(\text{compliers}|z) \quad (55.8)$$

$$Uplift = ATE_z = E_{1|1,z} - E_{1|0,z} \quad (55.9)$$

As we shall see, EU_z contains the information in PNS_z and ATE_z , plus much more.

One can find the stratum z^* such that $z^* = \operatorname{argmax} ATE_z$ (as is done A/B=RCT testing) or $z^* = \operatorname{argmax} EU_z$ or $z^* = \operatorname{argmax} PNS_z$. This is called the **unit selection problem**. It's called "unit selection" rather than "stratum selection" because once the stratum z^* is found, one can find a unit (i.e., individual) within that stratum.

55.1 Goal of PEU Theory

Everything that we said in Chapter 54 in the section entitled "Goal of PTE Theory" applies to PEU theory too, if we just replace PNS_z by EU_z . As in Chapter 54, we

will consider two types of bounds (for EU_z instead of PNS_z): (1) Bounds for an unspecified bnet, (2) Bounds for specific bnet families.

Here is an explanation of why EU_z varies within a bounded region, something that might not be obvious to the beginner. In Fig.55.1, we represent the utility function α_{y_0,y_1} by a unit vector $\hat{\alpha}$, the probability distribution $P_{y_0,y_1|z}$ by a vector \vec{P} , and EU_z by the dot product $\vec{P} \cdot \hat{\alpha}$. When the probability vector \vec{P} varies within a bounded region (shown in green), this causes the dot product $EU_z = \vec{P} \cdot \hat{\alpha}$ to vary within a bounded interval (also shown in green).

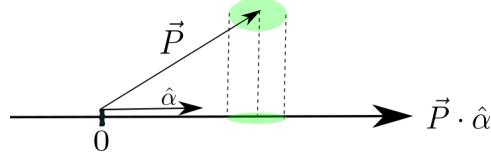


Figure 55.1: Bounds (shown in green) on the probability vector \vec{P} induce bounds (shown in green) on the dot product $EU_z = \vec{P} \cdot \hat{\alpha}$. Here $\hat{\alpha}$ is a unit vector that stands for a normalized utility function. $\hat{\alpha}$ does not vary but \vec{P} does.

55.2 Bnets for PEU Theory

Everything that we said in Chapter 54 in the section entitled “Bnets for PTE Theory” applies to PEU theory too, if we just replace PNS_z by EU_z .

55.3 Bounds on EU for unspecified bnet

Define the **balanced utility** by

$$\alpha_B = \alpha_{0,0} + \alpha_{1,1} , \quad (55.10)$$

the **unbalanced utility** by

$$\alpha_U = \alpha_{1,0} + \alpha_{0,1} , \quad (55.11)$$

and their difference by:

$$\sigma = \alpha_U - \alpha_B . \quad (55.12)$$

We will also use the abbreviations:

$$\alpha_{1-0,j} = \alpha_{1,j} - \alpha_{0,j} \quad (55.13)$$

and

$$\alpha_{j,1-0} = \alpha_{j,1} - \alpha_{j,0} . \quad (55.14)$$

Claim 81 *In general,*

$$\begin{cases} \max p_{[1\dots 4]} \leq EU_z \leq \min p_{[5\dots 8]} & \text{if } \sigma < 0 \\ \max p_{[5\dots 8]} \leq EU_z \leq \min p_{[1\dots 4]} & \text{if } \sigma > 0 \end{cases} \quad (55.15)$$

where³

$$\begin{cases} p_1 = \alpha_{0,1-0} E_{1|1,z} + \alpha_{j,0} E_{j|0,z} \\ p_2 = \alpha_{0-1,1} E_{0|0,z} + \alpha_{1,j} E_{j|1,z} \\ p_3 = p_5 + \sigma O_{*,*|z} \\ p_4 = p_1 - \sigma E_{1|0,z} + \sigma(1 - O_{*,*|z}) \end{cases} \quad (55.16)$$

$$\begin{cases} p_5 = \alpha_{1,1-0} E_{1|1,z} + \alpha_{j,0} E_{j|0,z} \\ p_6 = p_1 - \sigma E_{1|0,z} \\ p_7 = p_5 - \sigma E_{1|0,z} + \sigma P(\underline{y} = 1|z) \\ p_8 = p_1 - \sigma P(\underline{y} = 1|z) \end{cases} \quad (55.17)$$

proof: See Ref[22] or use the MRP (Matrix Representation of Probabilities) defined in Chapter 54.

QED

Later on, we will see that under monotonicity, these bounds collapse to a point estimate of EU_z . But what if monotonicity doesn't hold? In such cases, one can use the midpoint of the bounds as a non-rigorous point estimate of EU_z .

Claim 82 *In general,*

$$P_{0,0|z} = E_{0|1,z} - P_{1,0|z} \quad (55.18)$$

$$P_{1,1|z} = E_{1|0,z} - P_{1,0|z} \quad (55.19)$$

$$P_{0,1|z} = 1 - P_{0,0|z} - P_{1,1|z} - P_{1,0|z} \quad (55.20)$$

$$= 1 - E_{0|1,z} - E_{1|0,z} + P_{1,0|z} \quad (55.21)$$

proof:

$$P_{0,0|z} + P_{1,0|z} = P(\underline{y}_1 = 0|z) = E_{0|1,z} \quad (55.22)$$

³ $p_{[a\dots b]} = (p_a, p_{a+1}, \dots, p_b)$ for integers a, b such that $a < b$.

$$P_{1,1|z} + P_{1,0|z} = P(\underline{y}_0 = 1|z) = E_{1|0,z} \quad (55.23)$$

QED

Recall the definition of monotonicity. Monotonicity holds iff

$$P(\underline{y}_0 = 1, \underline{y}_1 = 0) = 0. \quad (55.24)$$

Claim 83 *In general,*

$$EU_z = \alpha_{0,0}P_{0,0|z} + \alpha_{1,1}P_{1,1|z} + \alpha_{0,1}P_{0,1|z} + \alpha_{1,0}P_{1,0|z} \quad (55.25)$$

Hence, if monotonicity holds, or $\alpha_{1,0} = 0$, then

$$EU_z = \alpha_{0,0}P_{0,0|z} + \alpha_{1,1}P_{1,1|z} + \alpha_{0,1}P_{0,1|z} \quad (55.26)$$

proof: Trivial

QED

Claim 83 is trivial, but it provides a good motivation and inspiration for the following less trivial claim.

Claim 84 *In general,*

$$EU_z = \alpha_{0,0} \underbrace{E_{0|1,z}}_{=1-E_{1|1,z}} + \alpha_{1,1}E_{1|0,z} + \alpha_{0,1} \underbrace{(1 - E_{0|1,z} - E_{1|0,z})}_{=E_{1|1,z}-E_{1|0,z}=ATE_z} + \sigma P_{1,0|z} \quad (55.27)$$

Hence, if monotonicity holds, or $\sigma = 0$, then

$$(EU_z)_{\sigma=0} = \alpha_{0,0}E_{0|1,z} + \alpha_{1,1}E_{1|0,z} + \alpha_{0,1}ATE_z \quad (55.28)$$

proof: In general,

$$EU_z = \begin{cases} \alpha_{0,0}P_{0,0|z} \\ + \alpha_{1,1}P_{1,1|z} \\ + \alpha_{0,1}P_{0,1|z} \\ + \alpha_{1,0}P_{1,0|z} \end{cases} = \begin{cases} \alpha_{0,0}(E_{0|1,z} - P_{1,0|z}) \\ + \alpha_{1,1}(E_{1|0,z} - P_{1,0|z}) \\ + \alpha_{0,1}(1 - E_{0|1,z} - E_{1|0,z} + P_{1,0|z}) \\ + \alpha_{1,0}P_{1,0|z} \end{cases} \quad (55.29)$$

Hence,

$$EU_z = (EU_z)_{\sigma=0} + \sigma P_{1,0|z} \quad (55.30)$$

QED

Perhaps this will make Claim 84 less mysterious to you. Note that Claims 83 and 84 imply the following:

$$\left[\frac{\partial EU_z}{\partial \alpha_{1,0}} \right]_{\alpha_{0,0}, \alpha_{1,1}, \alpha_{0,1}} = \left[\frac{\partial EU_z}{\partial \sigma} \right]_{\alpha_{0,0}, \alpha_{1,1}, \alpha_{0,1}} = P_{1,0|z} \quad (55.31)$$

55.4 Bounds on EU for specific bnet families

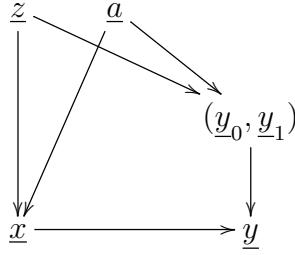


Figure 55.2: Bnet for $\underline{x} \rightarrow \underline{y}$ with 2 observed confounders $\underline{z}, \underline{a}$.

Fig.55.2 gives an example of a bnet that satisfies $(\underline{z} \cup \underline{a}) \cap de(\underline{x}) = \emptyset$.

Claim 85 *If $(\underline{z} \cup \underline{a}) \cap de(\underline{x}) = \emptyset$, then*

$$L \leq PNS_z \leq U \quad (\text{see Claim 72}) \quad (55.32)$$

$$\begin{cases} p_1 + \sigma U \leq EU_z \leq p_1 + \sigma L & \text{if } \sigma < 0 \\ p_1 + \sigma L \leq EU_z \leq p_1 + \sigma U & \text{if } \sigma > 0 \end{cases} \quad (55.33)$$

where

$$p_1 = \alpha_{0,1-0} E_{1|1,z} + \alpha_{j,0} E_{j|0,z} \quad (55.34)$$

$$L = \sum_a P(a|z) \max \left\{ \begin{array}{c} 0 \\ E_{1|1,a,z} - E_{1|0,a,z} \\ E_{0|0,a,z} - P(\underline{y} = 0|a, z) \\ E_{1|1,a,z} - P(\underline{y} = 1|a, z) \end{array} \right\} \quad (55.35)$$

$$U = \sum_a P(a|z) \min \left\{ \begin{array}{c} E_{1|1,a,z} \\ E_{0|0,a,z} \\ O_{*,*|a,z} \\ E_{*,*|a,z} - O_{*,*|a,z} \end{array} \right\} \quad (55.36)$$

proof:

QED

In Chapter 54, we defined what we mean when we say that \underline{z} is a partial or pure mediator relative to $(\underline{x}, \underline{y})$. Fig.55.3 is a copy of Fig.54.5.

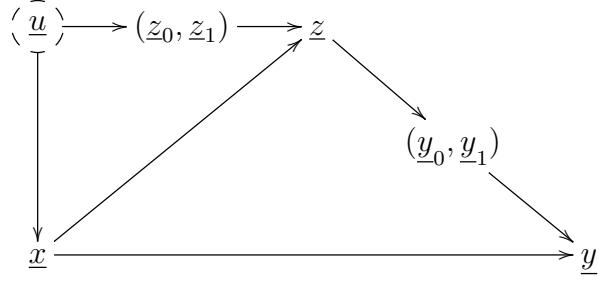


Figure 55.3: Example of a bnet in which \underline{z} is a partial mediator relative to $(\underline{x}, \underline{y})$. If \underline{z} is a partial mediator relative to $(\underline{x}, \underline{y})$, and, in addition, there is no arrow $\underline{x} \rightarrow \underline{y}$, we say \underline{z} is a pure mediator relative to $(\underline{x}, \underline{y})$.

Claim 86 (*Bounds if \underline{z} is partial mediator*)

If \underline{z} is a partial mediator relative to $(\underline{x}, \underline{y})$, and $\underline{x}, \underline{y}, \underline{z} \in \{0, 1\}$, then

$$L \leq PNS_z \leq U \quad (\text{see Claim 79}) \quad (55.37)$$

$$\begin{cases} p_1 + \sigma U \leq EU_z \leq p_1 + \sigma L & \text{if } \sigma < 0 \\ p_1 + \sigma L \leq EU_z \leq p_1 + \sigma U & \text{if } \sigma > 0 \end{cases} \quad (55.38)$$

where

$$p_1 = \alpha_{0,1-0} E_{1|1,z} + \alpha_{j,0} E_{j|0,z} \quad (55.39)$$

$$L = \max \left\{ \begin{array}{c} 0 \\ E_{1|1,z} - E_{1|0,z} \\ E_{0|0,z} - P(\underline{y} = 0|z) \\ E_{1|1,z} - P(\underline{y} = 1|z) \end{array} \right\} \quad (55.40)$$

$$U = \min \left\{ \begin{array}{c} E_{1|1,z} \\ E_{0|0,z} \\ O_{*,*|z} \\ E_{*,*|z} - O_{*,*|z} \\ new \end{array} \right\} \quad (55.41)$$

$$new = \sum_{z_0, z_1} \min \left\{ \begin{array}{c} O_{1|1,z_1,z} \\ O_{0|0,z_0,z} \end{array} \right\} \min \left\{ \begin{array}{c} P(z_1|z) \\ P(z_0|z) \end{array} \right\} \quad (55.42)$$

Claim 87 (*Bounds if \underline{z} is pure mediator*)

If \underline{z} is a pure mediator relative to $(\underline{x}, \underline{y})$, and $\underline{x}, \underline{y}, \underline{z} \in \{0, 1\}$, then Claim 86 for a partial mediator is still valid, except that now “new” equals

$$new = \sum_{z_0, z_1} \mathbb{1}(z_0 \neq z_1) \min \left\{ \begin{array}{l} O_{1|\underline{x}=z_1, z} \\ O_{0|\underline{x}=z_0, z} \end{array} \right\} \min \left\{ \begin{array}{l} P(\underline{z} = z_1 | \underline{x} = 1, z) \\ P(\underline{z} = z_0 | \underline{x} = 0, z) \end{array} \right\} \quad (55.43)$$

Chapter 56

Potential Outcomes and Beyond

This chapter is based on Ref.[6], a book by Stephen Cunningham entitled “Causal inference: the mixtape”.

The theory of potential outcomes (PO) was for the most part invented in a seminal 1974 paper by Donald B. Rubin. Rubin has also made important extensions to PO theory since 1974. However, he does not use Pearl’s causal DAGs to discuss PO theory. Pearl has shown that PO theory can be substantially clarified and extended by using the language of causal DAGs. The d-separation theorem and do operator that we discuss in Chapters 19 and 56 are especially useful in this regard. In this chapter, we stress the connection of PO theory to Pearl’s causal DAGs and bnets.

σ	d^σ	y^σ	$y^\sigma(0)$	$y^\sigma(1)$
Edith	0	5	5	.
Frank	0	7	7	.
George	0	8	8	.
Hank	0	10	10	.
Andy	1	10	.	10
Ben	1	5	.	5
Chad	1	16	.	16
Daniel	1	3	.	3

Table 56.1: PO dataset describing whether individual σ took a treatment drug ($d^\sigma = 1$) or didn’t ($d^\sigma = 0$). The treatment outcome is measured by the real number y^σ .

Suppose a **population of individuals** $\sigma = 0, 1, 2, \dots, nsam - 1$ is given ($d^\sigma = 1$) or is not given ($d^\sigma = 0$) a **treatment decision** d^σ , and that the **treatment outcome (i.e., response)** is measured by a real number y^σ . Table 56.1 gives a possible **PO dataset** for this scenario. As you can see from that table, each individual either takes a drug or doesn’t, but not both. PO theory can be viewed as a **missing data (MD) problem**. MD problems are discussed in Chapter 46. However, the PO MD problem is much more specialized than the generic MD problems discussed in Chapter 46. In the PO MD problem, we can fill in the blank cells by matching each

individual that took the drug with another *similar* individual that didn't. We will have much more to say about this matching strategy later in this chapter.

One can define similar individuals as individuals that have the same value for nx features $x^\sigma = (x_i^\sigma)_{i=0,1,\dots,nx-1}$. One can add to Table 56.1 nx extra columns giving the value of the feature vector x^σ for each individual. Members of a population with the same x^σ are referred to as a **subpopulation or stratum (i.e., layer)**.

In a **randomized controlled trial (RCT)**¹, the effect of the variable x^σ on the value of d^σ is eliminated by randomizing the population and therefore making the effect of x^σ on d^σ average out to zero. However, there are many situations in which carrying out an RCT is not possible or desirable. PO theory is a way of predicting the result of an RCT in situations where doing a real RCT is not possible or desirable.

In this chapter, x^σ will be called the confounders. Implicit throughout this chapter is the assumption that there are **no unmeasured confounders**. Because if there are some unmeasured confounders, those can send secret messages that influence the value that d^σ takes. This would ruin the predictions of someone trying to predict the results of an RCT without being privy to those secret messages. When there are **some unmeasured confounders**, it might still be possible to predict the effect of an RCT. This might be possible using instrumental variables. See Chapter 33 for a discussion of **instrumental variables**.

56.1 G and G_{den} bnets, the starting point bnets

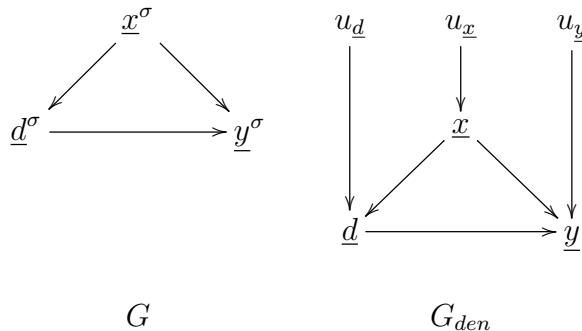


Figure 56.1: Bnets G and G_{den} are our starting point in discussing PO theory. G is for a single individual σ of the population. Bnet G_{den} is the DEN counterpart to G . DEN (Deterministic with External Noise) bnets are discussed in Chapter 38.

In this chapter, we will abbreviate $X[\sigma] = \underline{X}^\sigma$ for $X \in \{d, x, y\}$, where $\sigma \in \{0, 1, 2, \dots, nsam - 1\}$.

¹The term **A/B test** is often used to mean an RCT where A and B are the treated and control groups. However, sometimes the term is used to refer to an experiment that conditions on confounders, which violates the definition of an RCT, and is the same as a PO test.

For each individual (aka unit, sample) $\sigma = 0, 1, 2, \dots, nsam - 1$, let:
 $\underline{d}^\sigma \in \{0, 1\}$ be the treatment decision or drug dose. It equals 1 if treated and 0 if untreated.

$y^\sigma \in \mathbb{R}$ be the treatment potential outcome

\underline{x}^σ be the column vector of treatment confounders (aka covariates because they are often used as covariates (i.e., independent variables) in linear regression.)

Consider bnets G and G_{den} in Fig.56.1. G reflects the language used in Ref.[6] to discuss PO theory. And G_{den} reflects the language that Judea Pearl prefers to use to discuss PO theory. Both languages are equivalent. To go from one language to the other, one need only perform the following swaps, where \underline{u} is the external noise of the DEN bnet.

$$\underline{X}^\sigma \leftrightarrow \underline{X}(\underline{u}) \text{ for } X \in \{d, x, y\}.$$

$$P(\sigma) = \frac{1}{nsam} \leftrightarrow P(u)$$

$$\sum_\sigma P(\sigma)(\cdot) \leftrightarrow \sum_u P(u)(\cdot)$$

The TPMs, printed in blue, for the bnet G in Fig.56.1, are as follows:

$$P(x^\sigma) = P_{\underline{x}}(x^\sigma) \quad (56.1)$$

$$P(d^\sigma|x^\sigma) = P_{\underline{d}|\underline{x}}(d^\sigma|x^\sigma) \quad (56.2)$$

$$P(y^\sigma|d^\sigma, x^\sigma) = P_{\underline{y}|d,x}(y^\sigma|d^\sigma, x^\sigma) \quad (56.3)$$

Now let:

$\underline{d} \in \{0, 1\}$ be the treatment decision. It equals 1 if treated and 0 if untreated

$\underline{y} \in \mathbb{R}$ be the treatment potential outcome

\underline{x} be the column vector of treatment confounders (aka covariates)

$\underline{u} = (\underline{u}_d, \underline{u}_x, \underline{u}_y)$ be the external noise

The TPMs, printed in blue, for the bnet G_{den} in Fig.56.1, are as follows:

$$P(x|u_{\underline{x}}) = \mathbb{1}(x = u_{\underline{x}}) \quad (56.4)$$

$$P(d|x, u_d) = \mathbb{1}(d = f_d(x, u_d)) \quad (56.5)$$

$$P(y|d, x, u_y) = \mathbb{1}(y = f_y(d, x, u_y)) \quad (56.6)$$

If we linearize f_y in Eq.(56.6), we get

$$\underline{y} = \delta \underline{d} + \beta \underline{x} + \underline{u}_y , \quad (56.7)$$

where $\delta, \beta \in \mathbb{R}$. Assuming that $\underline{x}, \underline{y} \in \mathbb{R}$ and $\underline{d} \in \{0, 1\}$, Eq.(56.7) can be plotted. The resulting plot is given in Fig.56.2. This plot is a very special case of the PO problem, but it gives a crude idea of the “effects” $\delta = y(1) - y(0)$ that PO theory gives estimates for. Any individual participating in the experiment experiences either $y(1)$ or $y(0)$, but not both.

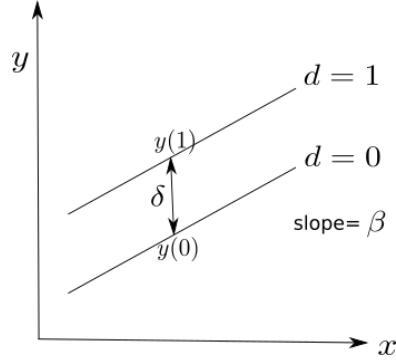


Figure 56.2: Plot of Eq.(56.7)

56.2 G bnet with nodes $y^\sigma(0), y^\sigma(1)$ added to it.

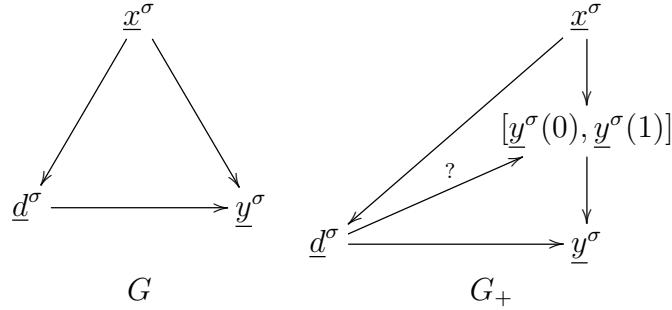


Figure 56.3: Bnet G_+ is bnet G with two new nodes $\underline{y}^\sigma(0)$ and $\underline{y}^\sigma(1)$ added to it. The tuple node $[\underline{y}^\sigma(0), \underline{y}^\sigma(1)]$ can also be represented by two nodes $\underline{c} \rightarrow \underline{y}(\underline{c})$, where $\underline{c} \in \{0, 1\}$.

Consider Fig.56.3. Bnet G_+ was obtained by adding two new nodes $\underline{y}^\sigma(0)$ and $\underline{y}^\sigma(1)$ to bnet G . The TPMs, printed in blue, for bnet G_+ , are as follows. Note that we define them in terms of the TPMs for bnet G .

$$P(x^\sigma) = P_{\underline{x}}(x^\sigma) \quad (56.8)$$

$$P(d^\sigma|x^\sigma) = P_{\underline{d}|\underline{x}}(d^\sigma|x^\sigma) \quad (56.9)$$

For $c \in \{0, 1\}$,

$$P(y^\sigma(c)|d^\sigma, x^\sigma) = \begin{cases} P_{\underline{y}(c)|\underline{d}, \underline{x}}(y^\sigma(c)|d^\sigma, x^\sigma) & \text{if INCLUDE arrow with question mark} \\ P_{\underline{y}(c)|\underline{x}}(y^\sigma(c)|x^\sigma) & \text{if EXCLUDE arrow with question mark} \end{cases} \quad (56.10)$$

$$P(y^\sigma|y^\sigma(0), y^\sigma(1), d^\sigma) = \mathbb{1}(y^\sigma = d^\sigma y^\sigma(1) + (1 - d^\sigma)y^\sigma(0)) \quad (56.11a)$$

$$= \mathbb{1}(y^\sigma = y^\sigma(d^\sigma)) \quad (56.11b)$$

Eq.(56.11) is often referred to as the **SUTVA assumption**.

If we sum over the nodes $\underline{y}(0)$ and $\underline{y}(1)$ of this bnet, we should get the bnet G . This is easy to check. Indeed,

$$P(y^\sigma|d^\sigma, x^\sigma) = \sum_{y^\sigma(0)} \sum_{y^\sigma(1)} \mathbb{1}(y^\sigma = y^\sigma(d^\sigma)) P(y^\sigma(0)|d^\sigma, x^\sigma) P(y^\sigma(1)|d^\sigma, x^\sigma) \quad (56.12)$$

$$= \begin{cases} P_{\underline{y}(0)|\underline{d}, \underline{x}}(y^\sigma|d^\sigma, x^\sigma) & \text{if } d^\sigma = 0 \\ P_{\underline{y}(1)|\underline{d}, \underline{x}}(y^\sigma|d^\sigma, x^\sigma) & \text{if } d^\sigma = 1 \end{cases}. \quad (56.13)$$

Henceforth, we will refer to the case where the question mark arrow is included as the general case, and to the case when it's excluded as the **weak-d limit**. Henceforth, we will first present results for the general case, and then describe how those results change for the weak-d limit. Rubinologists always assume the weak-d limit, but we find that with little effort, we can derive many results for general case, and then compare those results to their weak-d limit. I find such comparisons instructive.

Note that in the general case, $P(y(c) = y|\underline{d} = d, x)$ for $c, d \in \{0, 1\}$ are four different probability distributions, and that $P(\underline{y} = y|d = d, x)$ is defined in terms of two of them, the so called **factual distributions** with $c = d$. By measuring y , we can't access the other 2 probability distributions, the so called **counter-factual distributions** with $c \neq d$.

In the weak-d limit, $P(\underline{y}(c) = y|\underline{d} = d, x) = P(y(c) = y|x)$ are two probability distributions, and they both can be accessed by measuring y .

56.3 Expected Values of treatment outcome y^σ

It is convenient to define the following expected values of \underline{y}^σ in terms of the TPMs of bnet G_+ :

$$\mathcal{Y}_{c|d,x} = E_{\sigma|d,x}[\underline{y}^\sigma(c)] \rightarrow E_{\underline{y}(c)|d,x}[\underline{y}(c)] = \sum_y y P(\underline{y}(c) = y | \underline{d} = d, x) \quad (56.14)$$

$$\mathcal{Y}_{c|d} = E_{\sigma|d}[\underline{y}^\sigma(c)] \rightarrow E_{\underline{y}(c)|d}[\underline{y}(c)] = \sum_x \mathcal{Y}_{c|d,x} P(x|d) \quad (56.15)$$

$$\mathcal{Y}_{c|x} = E_{\sigma|x}[\underline{y}^\sigma(c)] \rightarrow E_{\underline{y}(c)|x}[\underline{y}(c)] = \sum_d \mathcal{Y}_{c|d,x} P(d|x) \quad (56.16)$$

$$\mathcal{Y}_c = E_\sigma[\underline{y}^\sigma(c)] \rightarrow E_{\underline{y}(c)}[\underline{y}(c)] = \sum_{x,d} \mathcal{Y}_{c|d,x} P(x, d) \quad (56.17)$$

Note that in the weak-d limit,

$$\mathcal{Y}_c = \sum_x \mathcal{Y}_{c|d,x} P(x) . \quad (56.18)$$

Note also that in the weak-d limit, $\mathcal{Y}_{c|d,x}$ is independent of d , but $\mathcal{Y}_{c|d}$ can depend on d if $P(x|d)$ depends on d .

$\mathcal{Y}_{0|0}, \mathcal{Y}_{1|1}$ are said to be **factual** (indicating compliant patients) whereas $\mathcal{Y}_{0|1}, \mathcal{Y}_{1|0}$ are said to be **counterfactual** (indicating non-compliant patients).

Also let

$$\mathcal{Y}_{|d,x} = E_{\sigma|d,x}[\underline{y}^\sigma] \rightarrow E_{\underline{y}|d,x}[\underline{y}] = \sum_y y P(\underline{y} = y | \underline{d} = d, x) \quad (56.19)$$

$$\mathcal{Y}_{|d} = E_{\sigma|d}[\underline{y}^\sigma] \rightarrow E_{\underline{y}|d}[\underline{y}] = \sum_x \mathcal{Y}_{|d,x} P(x|d) \quad (56.20)$$

$$\mathcal{Y}_{|x} = E_{\sigma|x}[\underline{y}^\sigma] \rightarrow E_{\underline{y}|x}[\underline{y}] = \sum_d \mathcal{Y}_{|d,x} P(d|x) \quad (56.21)$$

$$\mathcal{Y} = E_\sigma[\underline{y}^\sigma] \rightarrow E_{\underline{y}}[\underline{y}] = \sum_d \mathcal{Y}_{|d,x} P(d, x) \quad (56.22)$$

In the weak-d limit, $\mathcal{Y}_{|d,x}$ is independent of d , but $\mathcal{Y}_{|d}$ can still depend if $P(x|d)$ depends on d , then $\mathcal{Y}_{|d}$ depends on d too.

56.4 Translation Dictionary

Table 56.2 gives a dictionary for translating from the standard PO notation of Ref.[6] to our notation.

In standard PO notation	In our notation
i , individual (i.e., unit, sample) index	σ
$D_i = d_i$, treatment decision	$\underline{d}^\sigma = d^\sigma$
$Y_i = y_i$, treatment outcome	$\underline{y}^\sigma = y^\sigma$
$X_i = x_i$, treatment confounders	$\underline{x}^\sigma = x^\sigma$
$E[Y_i(c)]$	$E_\sigma[\underline{y}^\sigma(c)] = \mathcal{Y}_c$
$E[Y_i(c) D_i = d]$	$E_{\sigma d}[\underline{y}^\sigma(c)] = \mathcal{Y}_{c d}$
$E[Y_i(c) D_i = d, X_i = x]$	$E_{\sigma d,x}[\underline{y}^\sigma(c)] = \mathcal{Y}_{c d,x}$
$E[Y_i]$	$E_\sigma[\underline{y}^\sigma] = \mathcal{Y}$
$E[Y_i D_i = d]$	$E_{\sigma d}[\underline{y}^\sigma] = \mathcal{Y}_{ d}$
$E[Y_i D_i = d, X_i = x]$	$E_{\sigma d,x}[\underline{y}^\sigma] = \mathcal{Y}_{ d,x}$

Table 56.2: Dictionary for translating from standard PO notation of Ref.[6] to our notation. $c, d \in \{0, 1\}$.

56.5 $\mathcal{Y}_{|d,x} = \mathcal{Y}_{d|d,x}$ (SUTVA)

Claim 88 ²

$$\mathcal{Y}_{|d,x} = \mathcal{Y}_{d|d,x} \quad (56.23)$$

$$\mathcal{Y}_{|d} = \mathcal{Y}_{d|d} \quad (56.24)$$

proof:

$$\mathcal{Y}_{|d,x} = \sum_y y P(\underbrace{\underline{y}}_{=y(d) \text{ by Eq.(56.11)}} = y | d, x) \quad (56.25)$$

$$= \mathcal{Y}_{d|d,x}. \quad (56.26)$$

Applying $\sum_x P(x|d)$ to both sides of Eq.(56.23) gives Eq.(56.24).

QED

²In the standard PO notation, this is the frequently used identity

$$E[Y|D = d, x] = E[Y(d)|D = d, x]$$

.

56.6 Conditional Independence Assumption (CIA)

The **Conditional Independence Assumption** (CIA) is said to hold if

$$(\underline{y}^\sigma(0), \underline{y}^\sigma(1), \underline{y}^\sigma) \& \perp_P \underline{d}^\sigma | \underline{x}^\sigma . \quad (56.27)$$

This is satisfied by G_+ in the weak-d limit. To prove this, check that

$$(\underline{y}^\sigma(0), \underline{y}^\sigma(1), \underline{y}^\sigma) \perp_{G_+} \underline{d}^\sigma | \underline{x}^\sigma \quad (56.28)$$

and then invoke the d-separation theorem (see Chapter 19).

I think CIA only makes sense if the individuals are treatment blind (i.e., have no knowledge of whether they are in the treated or control groups.) Otherwise, that extra knowledge becomes a confounder not being included in x .

A **Randomized Controlled Trial (RCT)** is defined to satisfy Eq.(56.27) without the \underline{x}^σ conditioning; i.e., it satisfies

$$(\underline{y}^\sigma(0), \underline{y}^\sigma(1), \underline{y}^\sigma) \& \perp_P \underline{d}^\sigma . \quad (56.29)$$

This means that in an RCT, the arrow from \underline{x}^σ to \underline{d}^σ in G_+ is omitted.

Note that if we assume both CIA (i.e., weak-d limit) and SUTVA, we get

$$\mathcal{Y}_{|d,x} = E_{\sigma|d,x}[\underline{y}^\sigma] \quad (56.30a)$$

$$= E_\sigma[\underline{y}^\sigma | \underline{d}^\sigma = d, \underline{x}^\sigma = x] \quad (56.30b)$$

$$= E_\sigma[\underline{y}^\sigma(d) | \underline{d}^\sigma = d, \underline{x}^\sigma = x] \text{ (by SUTVA)} \quad (56.30c)$$

$$= E_\sigma[\underline{y}^\sigma(d) | \underline{x}^\sigma = x] \text{ (by CIA)} \quad (56.30d)$$

$$= E_{\sigma|x}[\underline{y}^\sigma(d)] \quad (56.30e)$$

$$= \mathcal{Y}_{d|x} . \quad (56.30f)$$

In an RCT, Eq.(56.30f) is valid without the x conditioning.

56.7 Treatment Effects

A **treatment effect** is a difference of two $\mathcal{Y}_{c|d}$. It is convenient to define the following treatment effects. See Figs.56.4 and 56.5.

- average treatment effect (ATE).

$$\textcolor{red}{ATE} = \mathcal{Y}_1 - \mathcal{Y}_0 = \delta \quad (56.31)$$

- average treatment effect of the treated (ATT)

$$\textcolor{red}{ATT} = \mathcal{Y}_{1|1} - \mathcal{Y}_{0|1} \quad (56.32)$$

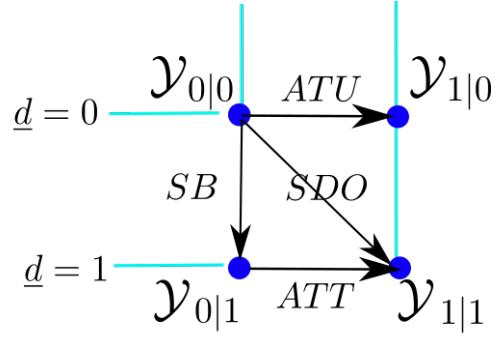


Figure 56.4: Different treatment effects. A treatment effect is a difference of two $\mathcal{Y}_{c|d}$.

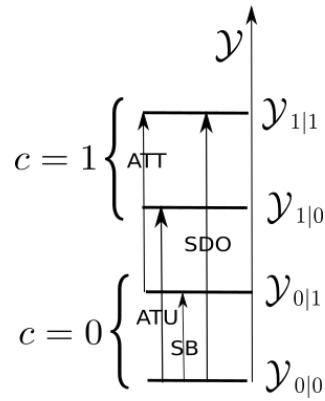


Figure 56.5: Alternative representation of the same information that is contained in Fig.56.4.

- average treatment effect of the untreated (ATU)

$$\textcolor{red}{ATU} = \mathcal{Y}_{1|0} - \mathcal{Y}_{0|0} \quad (56.33)$$

- selection bias (SB)

$$\textcolor{red}{SB} = \mathcal{Y}_{0|1} - \mathcal{Y}_{0|0} \quad (56.34)$$

- simple difference in outcomes (SDO)

$$\textcolor{red}{SDO} = \mathcal{Y}_{1|1} - \mathcal{Y}_{0|0} \quad (56.35)$$

Let

$$\pi_d = P(\underline{d} = d) \quad (56.36)$$

for $d \in \{0, 1\}$.

Note that there exist some linear constraints between these treatment effects.

$$\underbrace{\mathcal{Y}_1 - \mathcal{Y}_0}_{ATE} = \underbrace{\mathcal{Y}_{1|1}\pi_1 + \mathcal{Y}_{1|0}\pi_0}_{\mathcal{Y}_1} - \underbrace{(\mathcal{Y}_{0|1}\pi_1 + \mathcal{Y}_{0|0}\pi_0)}_{\mathcal{Y}_0} \quad (56.37)$$

$$= \underbrace{(\mathcal{Y}_{1|1} - \mathcal{Y}_{0|1})\pi_1}_{ATT} + \underbrace{(\mathcal{Y}_{1|0} - \mathcal{Y}_{0|0})\pi_0}_{ATU} \quad (56.38)$$

$$\underbrace{\mathcal{Y}_{1|1} - \mathcal{Y}_{0|0}}_{SDO} = \underbrace{(\mathcal{Y}_{1|1} - \mathcal{Y}_{0|1})}_{ATT} + \underbrace{\mathcal{Y}_{0|1} - \mathcal{Y}_{0|0}}_{SB} \quad (56.39)$$

$$\begin{aligned} \underbrace{\mathcal{Y}_{1|1} - \mathcal{Y}_{0|0}}_{SDO} &= \underbrace{(\mathcal{Y}_{1|1} - \mathcal{Y}_{0|1})\pi_1 + (\mathcal{Y}_{1|0} - \mathcal{Y}_{0|0})\pi_0}_{ATE} \\ &\quad + \underbrace{\mathcal{Y}_{0|1} - \mathcal{Y}_{0|0}}_{SB} \\ &\quad + \underbrace{(\mathcal{Y}_{1|1} - \mathcal{Y}_{0|1})}_{ATT}\pi_0 \\ &\quad - \underbrace{(\mathcal{Y}_{1|0} - \mathcal{Y}_{0|0})}_{ATU}\pi_0 \end{aligned} \quad (56.40)$$

By virtue of Eq.(56.38),

$$ATT = ATU \implies ATT = ATU = ATE \quad (56.41)$$

and

$$ATE = 0 \iff \frac{ATU}{ATT} = -\left(\frac{\pi_1}{\pi_0}\right). \quad (56.42)$$

Whenever $ATT = ATU$, we will say there is **T-U symmetry**.

In general, $SDO = ATT + SB$, but if there is T-U symmetry, then $SDO = ATE + SB$.

If there is T-U symmetry and zero bias $SB = 0$, then $SDO = ATE = ATT = ATU$.

If there is a null result for an RCT (i.e., $ATE = 0$), T-U symmetry and zero bias $SB = 0$, then $SDO = ATE = ATT = ATU = 0$.

Let

$$\mathcal{Y}_{c,d|x} = \mathcal{Y}_{c|d,x}P(d|x) \quad (56.43)$$

For each $\mathcal{E} \in \{ATE, ATT, ATU, SB, SDO\}$, we can define its restriction \mathcal{E}_x to a fixed stratum x by replacing each $\mathcal{Y}_{c|d}$ with $\mathcal{Y}_{c,d|x}$. For example,

$$ATT = \mathcal{Y}_{1|1} - \mathcal{Y}_{0|1} \text{ so } ATT_x = \mathcal{Y}_{1,1|x} - \mathcal{Y}_{0,1|x}. \quad (56.44)$$

We can calculate \mathcal{E} from \mathcal{E}_x using

$$\mathcal{E} = E_x[\mathcal{E}_x] = \sum_x P(x)\mathcal{E}_x . \quad (56.45)$$

56.8 Insights into what makes treatment effects equal and $\mathcal{Y}_{1|0} = \mathcal{Y}_1$

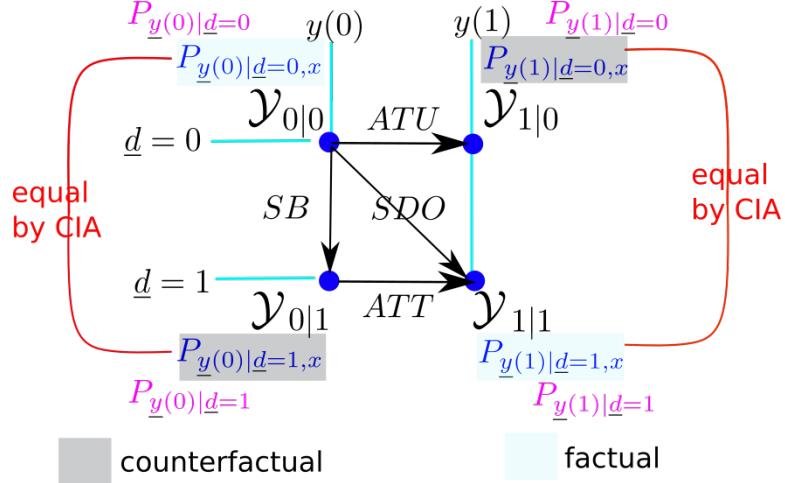


Figure 56.6: Figure 56.4 with added information about probability distributions used to obtain each expected value $\mathcal{Y}_{c|d}$.

1. Is it possible for $SDO = 0$ but $ATE \neq 0$ or vice versa, and what is going on when this is true?
2. What is going on when two treatment effects are equal; for instance, when $ATT = ATU$?
3. When is $\mathcal{Y}_{1|0} = \mathcal{Y}_1$, and what is going on when this is true?

Fig.56.6 gives some intuition about what is going on when any of these things happen.

Recall that each expected value $\mathcal{Y}_{c|d}$ is associated with a probability distribution $P_{\underline{y}(c)|\underline{d},x}$.

$$\mathcal{Y}_{c|d} = \sum_y y \underbrace{\sum_x P_{\underline{y}(c)|\underline{d},x}(y|d, x)P(x|d)}_{P_{\underline{y}(c)|\underline{d}}(y|d)} \quad (56.46)$$

for $c, d \in \{0, 1\}$. Fig.56.6 reminds us of which P is used to generate each \mathcal{Y} . From this figure, we see that

1. A sufficient condition for $SDO = 0$ is that $P_{\underline{y}(1)|\underline{d}=1} = P_{\underline{y}(0)|\underline{d}=0}$. A sufficient condition for $ATE = 0$ is that $P_{\underline{y}(1)|x} = P_{\underline{y}(0)|x}$.
2. A sufficient condition for $ATT = ATU$ is that $P_{\underline{y}(1)|\underline{d}=1,x} - P_{\underline{y}(0)|\underline{d}=1,x}$ equals $P_{\underline{y}(1)|\underline{d}=0,x} - P_{\underline{y}(0)|\underline{d}=0,x}$.
3. A sufficient condition for $\mathcal{Y}_{1|0} = \mathcal{Y}_1$ is that $P_{\underline{y}(1)|\underline{d}=0} = P_{\underline{y}(1)}$. Note that the CIA implies that $P_{\underline{y}(1)|\underline{d}=0,x} = P_{\underline{y}(1)|x}$ always, but this does not imply that $P_{\underline{y}(1)|\underline{d}=0} = P_{\underline{y}(1)}$.

56.9 G_{do+} bnet

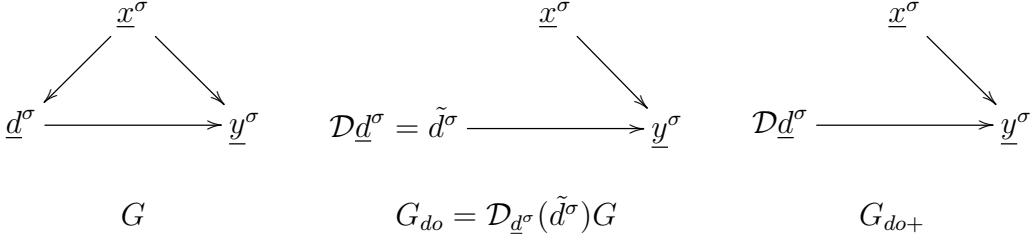


Figure 56.7: Bnet $G_{do} = \mathcal{D}_{\underline{d}^\sigma}(\tilde{d}^\sigma)G$ is obtained by applying the do operator to node \underline{d}^σ of bnet G . Bnet G_{do+} is obtained by adding a prior probability distribution $P(\underline{d}^\sigma)$ to node $\mathcal{D}\underline{d}^\sigma$ of bnet G_{do} .

Fig.56.7 shows how bnet G_{do} is obtained by applying the do operator to bnet G , and how bnet G_{do+} is obtained by adding a prior probability distribution to one of the nodes of G_{do} . In bnet G_{do} , node \underline{d}^σ has been stripped of all outside influences and fixed to a specific state \tilde{d}^σ . This is what an RCT does.

The TPMs, printed in blue, for the bnets G_{do} and G_{do+} , are as follows. Note that the TPMs for bnets G_{do} and G_{do+} are defined in terms of the TPMs of bnet G .

$$P(x^\sigma) = P_{\underline{x}}(x^\sigma) \quad (56.47)$$

$$P_{\mathcal{D}\underline{d}}(\tilde{d}) = \sum_x P_{\underline{d}|\underline{x}}(\tilde{d}|x)P_{\underline{x}}(x) \quad (56.48)$$

$$P(\tilde{d}^\sigma) = \begin{cases} \delta(\underline{\tilde{d}^\sigma}, \tilde{d}^\sigma) & \text{for } G_{do} \\ P_{\mathcal{D}\underline{d}}(\tilde{d}^\sigma) & \text{for } G_{do+} \end{cases} \quad (56.49)$$

$$P(y^\sigma | \tilde{d}^\sigma, x^\sigma) = P_{\underline{y}|\underline{d}, \underline{x}}(y^\sigma | \tilde{d}^\sigma, x^\sigma) \quad (56.50)$$

Note that in G_{do} ,

$$P(\underline{y} = y | \mathcal{D}\underline{d} = d, \underline{x} = x) = P(y | \underline{d} = d, x) \quad (56.51)$$

because, by the d-separation theorem, when we condition on the confounder \underline{x} , we block information from being transmitted from \underline{d} to \underline{y} through \underline{x} , and this is equivalent to amputating the arrow $\underline{x} \rightarrow \underline{d}$.

Using Eq.(56.51), we get

$$P(\underline{y} = y | \mathcal{D}\underline{d} = d) = \sum_x P(\underline{y} = y | \mathcal{D}\underline{d} = d, x) P(x | \mathcal{D}\underline{d} = d) \quad (56.52)$$

$$= \sum_x P(y | d, x) P(x) \quad (56.53)$$

Eq.(56.53) is called the **backdoor adjustment formula**. It allows us to express a probability with a do operator in its definition in terms of probabilities without do operators.

56.10 $ACE = ATE$

Define the Average Causal Effect (ACE) by

$$ACE = \sum_y y [P(y | \mathcal{D}\underline{d} = 1) - P(y | \mathcal{D}\underline{d} = 0)] \quad (56.54)$$

$$= \sum_x P(x) \sum_y y [P(y | \underline{d} = 1, x) - P(y | \underline{d} = 0, x)] . \text{ (by Eq.(56.53))} \quad (56.55)$$

Claim 89 *If we assume both SUTVA and CIA (i.e., weak-d limit), then*

$$ACE = ATE \quad (56.56)$$

proof:

$$ACE = \sum_x P(x) \sum_y y [P(\underline{y} = y | \underline{d} = 1, x) - P(\underline{y} = y | \underline{d} = 0, x)] \quad (56.57)$$

$$= \sum_x P(x) [\mathcal{Y}_{1|x} - \mathcal{Y}_{0|x}] \text{ (by SUTVA)} \quad (56.58)$$

$$= \sum_x P(x) [\mathcal{Y}_{1|x} - \mathcal{Y}_{0|x}] \text{ (by CIA)} \quad (56.59)$$

$$= \mathcal{Y}_1 - \mathcal{Y}_0 \quad (56.60)$$

$$= ATE \quad (56.61)$$

QED

We will say there is a **null result in an RCT** when $ACE = 0$. By the previous claim, this is true iff $ATE = 0$ (assuming weak-d limit).

56.11 Good, Bad Controls

The bnet G_+ of Fig.56.3, —the cornerstone of Rubin’s PO theory— is limited in scope and is easily misapplied, leading to incorrect results. The problem is some features that are available and could be conditioned on shouldn’t because they would introduce spurious contributions to ATE. Such features are called “bad controls”.

Examples:

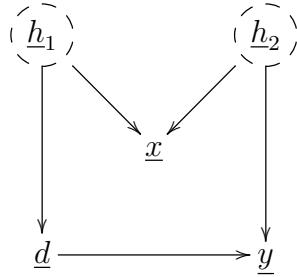


Figure 56.8: In this bnet, \underline{x} is a bad control (i.e., should not be conditioned on). Nodes \underline{h}_1 and \underline{h}_2 are hidden and therefore cannot be conditioned on.

1. Consider the bnet Fig.56.8, which Pearl calls M-bias, because it looks like an M. In that figure, \underline{x} is a “bad control” because calculating ATE by conditioning on it, and using the formula $ATE = \sum_x P(x)[\mathcal{Y}_{1|x} - \mathcal{Y}_{0|x}]$, yields a value of ATE that is different from ACE . This value for ATE is unacceptable because it does not give the result of an RCT whereas ACE defined in terms of do operators

always does. The reason³ $ATE \neq ACE$ for this figure is that in it, \underline{x} is a collider node, and conditioning on it allows rather than prevents information to flow from \underline{d} to \underline{y} via the path $\underline{d} - \underline{h}_1 - \underline{x} - \underline{h}_2 - \underline{y}$.

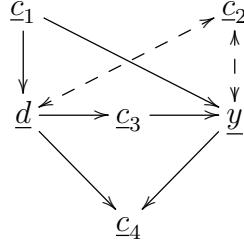


Figure 56.9: In this bnet, node c_1 is a good control and nodes c_2, c_3, c_4 are bad ones.

2. In Fig.56.9, node c_1 is a good control and nodes c_2, c_3, c_4 are bad ones.

Conditioning on c_1 blocks path $\underline{d} - c_1 - \underline{y}$, good

Conditioning on c_2 opens path $\underline{d} - c_2 - \underline{y}$, bad

Conditioning on c_3 blocks path $\underline{d} - c_3 - \underline{y}$, bad

Conditioning on c_4 opens path $\underline{d} - c_4 - \underline{y}$, bad

Pearl et al have a paper (Ref.[5]) that I highly recommend that gives dozens of examples of good, neutral and bad controls in an ACE calculation.

56.12 PO Confounder Sensitivity Analysis

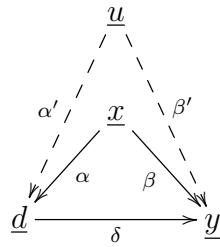


Figure 56.10: LDEN bnet used to do PO confounder sensitivity analysis. Node u is an unobserved confounder.

Consider the LDEN bnet of Fig.56.10. whose structure equations, printed in blue, are as follows:

³We are using here arguments based on the d-separation theorem which is discussed in Chapter 19.

$$\underline{d} = \alpha \underline{x} + \alpha' \underline{u} \quad (56.62)$$

$$\underline{y} = \beta \underline{x} + \beta' \underline{u} + \delta \underline{d} \quad (56.63)$$

Therefore,

$$\underline{u} = \frac{1}{\alpha'} (\underline{d} - \alpha \underline{x}) \quad (56.64)$$

$$\underline{y} = \left(\beta - \frac{\alpha \beta'}{\alpha'} \right) \underline{x} + \left(\delta + \frac{\beta'}{\alpha'} \right) \underline{d} \quad (56.65)$$

$$\mathcal{Y}_{d|x,u} = E[\underline{y}(d)|x, u] = E[\underline{y}|d, x, u] = \beta x + \beta' u + \delta d \quad (56.66)$$

$$\mathcal{Y}_{d|x} = E[\underline{y}(d)|x] = E[\underline{y}|d, x] = \left(\beta - \frac{\alpha \beta'}{\alpha'} \right) x + \left(\delta + \frac{\beta'}{\alpha'} \right) d \quad (56.67)$$

$$ATE = E_x[ATE_x] = E_x[\mathcal{Y}_{1|x} - \mathcal{Y}_{0|x}] = \delta + \frac{\beta'}{\alpha'} \quad (56.68)$$

$$ATE|_{\beta'=0} = \delta = \frac{dy}{d\underline{d}} \quad (56.69)$$

$$ATE - ATE|_{\beta'=0} = \frac{\beta'}{\alpha'} = -\frac{\frac{dy}{d\underline{u}}}{\frac{d\underline{d}}{d\underline{u}}} \quad (56.70)$$

56.13 (SDO, ATE) space

If we substitute $y^\sigma \rightarrow y^\sigma(d^\sigma)$ and $y^{m(\sigma)} \rightarrow y^\sigma(1 - d^\sigma)$ into the estimator Eq.(56.83) for ATE and the estimator Eq.(56.89) for SDO , we get

$$\widehat{ATE}_x = \frac{1}{N_x} \sum_{\sigma \in A_x} (2d^\sigma - 1)[y^\sigma(d^\sigma) - y^\sigma(1 - d^\sigma)] \quad (56.71)$$

$$= \frac{1}{N_x} \sum_{\sigma \in A_x} [y^\sigma(1) - y^\sigma(0)] \quad (56.72)$$

and

$$\widehat{SDO}_x = \frac{1}{N_{1,x}} \sum_{\sigma \in A_x} d^\sigma y^\sigma(d^\sigma) - \frac{1}{N_{0,x}} \sum_{\sigma \in A_x} (1 - d^\sigma) y^\sigma(d^\sigma) \quad (56.73)$$

$$= \frac{1}{N_{1,x}} \sum_{\sigma \in A_{1,x}} y^\sigma(1) - \frac{1}{N_{0,x}} \sum_{\sigma \in A_{0,x}} y^\sigma(0). \quad (56.74)$$

Recall that $\hat{\mathcal{E}} = E_x[\hat{\mathcal{E}}_x] = \sum_x \frac{N_x}{N} \hat{\mathcal{E}}_x$ for $\mathcal{E} \in \{ATE, SDO\}$.

Recall also that $ACE = ATE = 0$ is the null result in an RCT.

Suppose that the treatment outcome y^σ has only two possible values, 0 and 1. Then, $-1 \leq ATE \leq 1$ and $-1 \leq SDO \leq 1$. But does $ATE = 0$ imply $SDO = 0$ or vice versa? Next, we answer that question and more by finding the region of accessibility in the (SDO, ATE) plane, assuming $y^\sigma \in \{0, 1\}$.

σ	d^σ	$y^\sigma(0)$	$y^\sigma(1)$
1	0	1	0
2	0	1	0
3	0	1	0
4	1	1	0
5	1	1	0
6	1	1	0

(a) $ATE = -1$ ($SDO = -1$)
point A

σ	d^σ	$y^\sigma(0)$	$y^\sigma(1)$
1	0	0	1
2	0	0	1
3	0	0	1
4	1	0	0
5	1	0	0
6	1	0	0

(b) $ATE = \frac{1}{2}$ ($SDO = 0$)
point B

σ	d^σ	$y^\sigma(0)$	$y^\sigma(1)$
1	0	0	1
2	0	0	1
3	0	0	1
4	1	0	1
5	1	0	1
6	1	0	1

(c) $ATE = 1$ ($SDO = 1$)
point C

Figure 56.11: Examples of PO datasets. Exploring ATE extremes.

σ	d^σ	$y^\sigma(0)$	$y^\sigma(1)$
1	0	1	1
2	0	1	1
3	0	1	1
4	1	0	0
5	1	0	0
6	1	0	0

(a) $SDO = -1$ ($ATE = 0$)
point D

σ	d^σ	$y^\sigma(0)$	$y^\sigma(1)$
1	0	1	0
2	0	1	0
3	0	1	0
4	1	1	1
5	1	1	1
6	1	1	1

(b) $SDO = 0$ ($ATE = -\frac{1}{2}$)
point E

σ	d^σ	$y^\sigma(0)$	$y^\sigma(1)$
1	0	0	0
2	0	0	0
3	0	0	0
4	1	1	1
5	1	1	1
6	1	1	1

(c) $SDO = 1$ ($ATE = 0$)
point F

Figure 56.12: Examples of PO datasets. Exploring SDO extremes.

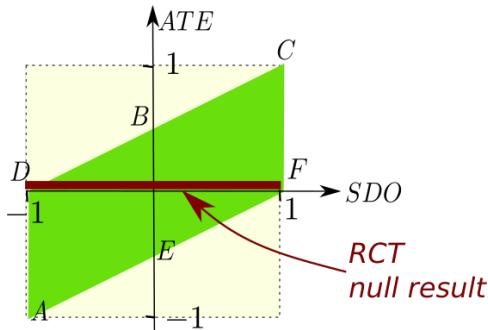


Figure 56.13: Green parallelogram is accessible region in (SDO, ATE) plane, assuming $y^\sigma \in \{0, 1\}$. Each of the six points A, B, ... F corresponds to one of the six tables in Figs. 56.11 and 56.12. Segment DF corresponds to the null result in an RCT.

56.14 Strata-Matching

For a situation described by the bnet G_+ in the weak-d limit, we can match *similar* individuals to fill the blank cells of Table 56.1. By “similar”, we mean that they have the same or almost the same value of x^σ .

The reason the weak-d limit is required is because it implies that $P(y(c)|d = 0, x) = P(y(c)|d = 1, x)$ for $c \in \{0, 1\}$. Hence, we can sample from a known factual ($c = d$) distribution to fill the missing data in the unknown counterfactual ($c \neq d$) distribution.

56.14.1 Exact strata-matching

Estimates of Treatment Effects

For $d \in \{0, 1\}$ and all strata x , define the sets of individuals $A_{d,x} = \{\sigma : d^\sigma = d, x^\sigma = x\}$, $A_x = A_{0,x} \cup A_{1,x}$ and $A = \cup_x A_x$. Let $N_{d,x} = |A_{d,x}|$, $N_x = |A_x|$ and $N = |A|$.

In an exact strata-matching, we match each individual with $d^\sigma = d, x^\sigma = x$ with exactly one individual with $d^\sigma = 1 - d, x^\sigma = x$. Define a map $m : A \rightarrow A$ such that, for each x , and for $d \in \{0, 1\}$, if $\sigma \in A_{d,x}$, then $m(\sigma) \in A_{1-d,x}$. This assumes $A_{0,x}$ and $A_{1,x}$ are non-empty for all x . The purpose of map $m()$ is to fill in the missing data in the PO dataset. See Fig.56.3 for a pictorial representation of this.

	$y^\sigma(0)$	$y^\sigma(1)$
$d^\sigma = 0$	y^σ	$y^{m(\sigma)}$
$d^\sigma = 1$	$y^{m(\sigma)}$	y^σ

Table 56.3: Illustration of the purpose of the map $m()$. Note that $y^\sigma = y^\sigma(d^\sigma)$ and $y^{m(\sigma)} = y^\sigma(1 - d^\sigma)$.

Note that

$$\sum_{\sigma \in A_x} \frac{d^\sigma}{N_{1,x}} = \sum_{\sigma \in A_{1,x}} \frac{1}{N_{1,x}} = 1. \quad (56.75)$$

Thus

$$\sum_{\sigma \in A_x} \frac{d^\sigma}{N_{1,x}} y^\sigma = E_{\sigma|d=1,x}[y^\sigma] = \mathcal{Y}_{1|x} \quad (56.76)$$

Table 56.4 gives estimates of $\mathcal{Y}_{c|d,x}$

Recall that

$$\mathcal{Y}_{c,d|x} = \mathcal{Y}_{c|d,x} P(d|x) \quad (56.77)$$

Hence,

	$y^\sigma(0)$	$y^\sigma(1)$
$d^\sigma = 0$	$\frac{1}{N_{0,x}} \sum_{\sigma \in A_x} (1 - d^\sigma) y^\sigma = \mathcal{Y}_{0 0,x}$	$\frac{1}{N_{0,x}} \sum_{\sigma \in A_x} (1 - d^\sigma) y^{m(\sigma)} = \mathcal{Y}_{1 0,x}$
$d^\sigma = 1$	$\frac{1}{N_{1,x}} \sum_{\sigma \in A_x} d^\sigma y^{m(\sigma)} = \mathcal{Y}_{0 1,x}$	$\frac{1}{N_{1,x}} \sum_{\sigma \in A_x} d^\sigma y^\sigma = \mathcal{Y}_{1 1,x}$

Table 56.4: Estimates of $\mathcal{Y}_{c|d,x}$.

	$y^\sigma(0)$	$y^\sigma(1)$
$d^\sigma = 0$	$\frac{1}{N_x} \sum_{\sigma \in A_x} (1 - d^\sigma) y^\sigma = \mathcal{Y}_{0,0 x}$	$\frac{1}{N_x} \sum_{\sigma \in A_x} (1 - d^\sigma) y^{m(\sigma)} = \mathcal{Y}_{1,0 x}$
$d^\sigma = 1$	$\frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma y^{m(\sigma)} = \mathcal{Y}_{0,1 x}$	$\frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma y^\sigma = \mathcal{Y}_{1,1 x}$

Table 56.5: Estimates of $\mathcal{Y}_{c,d|x}$.

$$\mathcal{Y}_{c,d|x} = (N_{d,x} \mathcal{Y}_{c|d,x}) \frac{P(d|x)}{N_{d,x}} \quad (56.78)$$

$$= (N_{d,x} \mathcal{Y}_{c|d,x}) \frac{1}{N_x} \quad (56.79)$$

Table 56.5 gives estimates of $\mathcal{Y}_{c,d|x}$

The treatment effects $\mathcal{E} \in \{ATE, ATT, ATU, SB, SDO\}$ can be estimated from the data via the following estimators.

$$\widehat{ATE}_x = \overbrace{\mathcal{Y}_{1|1,x} P(1|x) + \mathcal{Y}_{1|0,x} P(0|x)}^{\mathcal{Y}_{1|x}} - \overbrace{\mathcal{Y}_{0|1,x} P(1|x) + \mathcal{Y}_{0|0,x} P(0|x)}^{\mathcal{Y}_{0|x}} \quad (56.80)$$

$$= \frac{1}{N_x} [\widehat{ATT}_x N_{1,x} + \widehat{ATU}_x N_{0,x}] \quad (56.81)$$

$$= \frac{1}{N_x} \left[\sum_{\sigma \in A_x} d^\sigma [y^\sigma - y^{m(\sigma)}] + \sum_{\sigma \in A_x} (1 - d^\sigma) [y^{m(\sigma)} - y^\sigma] \right] \quad (56.82)$$

$$= \frac{1}{N_x} \sum_{\sigma \in A_x} (2d^\sigma - 1) [y^\sigma - y^{m(\sigma)}] \quad (56.83)$$

$$\widehat{ATT}_x = \overbrace{\frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma y^\sigma}^{\mathcal{Y}_{1,1|x}} - \overbrace{\frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma y^{m(\sigma)}}^{\mathcal{Y}_{0,1|x}} \quad (56.84)$$

$$= \frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma [y^\sigma - y^{m(\sigma)}] \quad (56.85)$$

$$\widehat{ATU}_x = \overbrace{\frac{1}{N_x} \sum_{\sigma \in A_x} (1 - d^\sigma) y^{m(\sigma)}}^{\mathcal{Y}_{1,0|x}} - \overbrace{\frac{1}{N_x} \sum_{\sigma \in A_x} (1 - d^\sigma) y^\sigma}^{\mathcal{Y}_{0,0|x}} \quad (56.86)$$

$$= \frac{1}{N_x} \sum_{\sigma \in A_x} (1 - d^\sigma) [y^{m(\sigma)} - y^\sigma] \quad (56.87)$$

$$\widehat{SB}_x = \overbrace{\frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma y^{m(\sigma)}}^{\mathcal{Y}_{0,1|x}} - \overbrace{\frac{1}{N_x} \sum_{\sigma \in A_x} (1 - d^\sigma) y^\sigma}^{\mathcal{Y}_{0,0|x}} \quad (56.88)$$

$$\widehat{SDO}_x = \overbrace{\frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma y^\sigma}^{\mathcal{Y}_{1,1|x}} - \overbrace{\frac{1}{N_x} \sum_{\sigma \in A_x} (1 - d^\sigma) y^\sigma}^{\mathcal{Y}_{0,0|x}} \quad (56.89)$$

Suppose we do linear regression to fit a hyperplane $y(x)$ to the dataset set $\{(x^\sigma, y^\sigma) : \sigma\}$, and then we calculate $\frac{dy}{dd} = \delta$. Out of all the treatment effects, this δ is probably (?) closest to $ACE = ATE$. Note also that the linear regression method of estimating δ does imputation (guesses missing values) by doing a linear fit. One can also use machine learning to do a non-linear fit. In contrast, the estimators of treatment effects presented in this section do imputation by non-linear strata-matching.

Example, estimation of treatment effects

For $\sigma \in \{1, 2, \dots, 10\}$, define

$$m(\sigma) = \begin{cases} \sigma + 5 & \text{if } \sigma \leq 5 \\ \sigma - 5 & \text{if } \sigma > 5 \end{cases} \quad (56.90)$$

Let $N(\mathcal{S})$ be the number of individuals σ that satisfy condition \mathcal{S} . For example, $N(\underline{d^\sigma} = d)$ is the number of individuals such that $\underline{d^\sigma} = d$.

$$N_1 = N(d^\sigma = 1) = 5 \quad (56.91)$$

$$N_0 = N(d^\sigma = 0) = 5 \quad (56.92)$$

$$N = N_0 + N_1 = 10 \quad (56.93)$$

$$\mathcal{Y}_{1|1} = \frac{1}{N_1} \sum_{\sigma} d^\sigma y^\sigma = \frac{4}{5} \quad (56.94)$$

σ	d^σ	y^σ	$d^\sigma y^\sigma$	$(1 - d^\sigma)y^\sigma$	$d^\sigma y^{m(\sigma)}$	$(1 - d^\sigma)y^{m(\sigma)}$
1	0	0	0	0	0	0
2	0	0	0	0	0	1
3	0	1	0	1	0	1
4	0	1	0	1	0	1
5	0	1	0	1	0	1
6	1	0	0	0	0	0
7	1	1	1	0	0	0
8	1	1	1	0	1	0
9	1	1	1	0	1	0
10	1	1	1	0	1	0

Table 56.6: Estimators of treatment effects are calculated for this example.

$N(d, y)$	$y = 0$	$y = 1$
$d = 0$	2	3
$d = 1$	1	4

Table 56.7: $N(\underline{d}^\sigma = d, \underline{y}^\sigma = y)$ for the data in Table 56.6.

$$\mathcal{Y}_{0|0} = \frac{1}{N_0} \sum_{\sigma} (1 - d^\sigma) y^\sigma = \frac{3}{5} \quad (56.95)$$

$$\mathcal{Y}_{0|1} = \frac{1}{N_1} \sum_{\sigma} d^\sigma y^{m(\sigma)} = \frac{3}{5} \quad (56.96)$$

$$\mathcal{Y}_{1|0} = \frac{1}{N_0} \sum_{\sigma} (1 - d^\sigma) y^{m(\sigma)} = \frac{4}{5} \quad (56.97)$$

$$ATT = \mathcal{Y}_{1|1} - \mathcal{Y}_{0|1} = \frac{1}{5} \quad (56.98)$$

$$ATE = ATT = ATU = SDO = \frac{1}{5}, \quad SB = 0 \quad (56.99)$$

This example is unusual in that it has a single stratum x , and for that stratum, the treated and untreated populations are **balanced** (of equal size). Also, the map $m()$ is 1-1 onto. If, for instance, $m(\sigma) = 6$ for all $\sigma \in A_0$ and $m(\sigma) = 5$ for $\sigma \in A_1$, then ATE, ATT, ATU, SDO would not all be same, and SB would not be zero.

In fact, whenever there is a single balanced stratum and the map $m()$ is 1-1 onto, Eq.(56.99) can be proven to be true using the methods of section 56.8.

56.14.2 Approximate strata-matching

It is very often the case that one can't find for a given individual σ another individual that has opposite d^σ but exactly the same value of x^σ . In such cases, one can discard all matchless individuals. But that would entail a loss of precious information. Instead of discarding orphans, a better way is to relax our demands and match individual σ with another individual $m(\sigma)$ such that x^σ and $x^{m(\sigma)}$ are very close in some metric. Alternatively, the matching individual might not be real; it might be a composite of individuals.

More precisely, for some arbitrary parameter $\epsilon > 0$, and an individual σ , define the **strata-matching set** $\mathcal{M}_\epsilon(\sigma)$ by⁴

$$\mathcal{M}_\epsilon(\sigma) = \{m : d^m = 1 - d^\sigma, \text{dist}(x^\sigma, x^m) \leq \epsilon\}, \quad (56.100)$$

where

$$\text{dist}(x^\sigma, x^m) = [x^\sigma]^T [\Sigma]^{-1} x^m, \quad (56.101)$$

where $\Sigma = \langle x^\sigma, [x^m]^T \rangle$. This metric $\text{dist}(x^\sigma, x^m)$ is called the **Mahalanobis distance**. We will call the case $\epsilon = 0$ an **exact strata-matching**, and the case $\epsilon \neq 0$ an **approximate strata-matching**. To do an approximate strata-matching, replace $y^{m(\sigma)}$ by $\langle y \rangle^{\mathcal{M}_\epsilon(\sigma)}$ in the estimators given above for an exact strata-matching. $\langle y \rangle^{\mathcal{M}_\epsilon(\sigma)}$ is defined by

$$\langle y \rangle^{\mathcal{M}_\epsilon(\sigma)} = \frac{1}{|\mathcal{M}_\epsilon(\sigma)|} \sum_{m \in \mathcal{M}_\epsilon(\sigma)} y^m. \quad (56.102)$$

56.14.3 Unbiased strata-matching estimators

The estimators we obtained via strata-matching are biased because strata-matching, due to its non-uniqueness, introduces noise into the estimate. However, one can define new bias-corrected estimators. Following Ref.[6], we will next find an unbiased estimator of ATT_x using the biased estimator of ATT_x that we obtained by strata-matching.

Let $\hat{\mathcal{Y}}_{|d,x}$ be an estimate of $\mathcal{Y}_{|d,x} = E_{|d,x}[y]$ that is obtained, for example, via Linear Regression.

Claim 90 *The quantity*

⁴One can use an ϵ that depends on σ . For example, let $\epsilon(\sigma, 5)$ satisfy $|\mathcal{M}_{\epsilon(\sigma, 5)}(\sigma)| = 5$.

$$\widehat{ATT}_x^{unbi} = \frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma \left[(y^\sigma - y^{m(\sigma)}) - (\widehat{\mathcal{Y}}_{|0,x^\sigma} - \widehat{\mathcal{Y}}_{|0,x^{m(\sigma)}}) \right] \quad (56.103)$$

is an unbiased estimate of ATT_x .

proof:

We begin by assuming a special case of SUTVA. Let

$$\underline{y}^\sigma = \underline{y}(d^\sigma) = \widehat{\mathcal{Y}}_{|d^\sigma,x^\sigma} + \underline{\epsilon}^\sigma \quad (56.104)$$

where

$$\left\langle \widehat{\mathcal{Y}}_{|d^\sigma,x^\sigma}, \underline{\epsilon}^\sigma \right\rangle_\sigma = 0 \quad (56.105)$$

Recall that the biased estimator of ATT_x obtained by strata-matching is

$$\widehat{ATT}_x^{bi} = \frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma (y^\sigma - y^{m(\sigma)}) \quad (56.106)$$

where σ and $m(\sigma)$ are matched (i.e., $x^\sigma \approx x^{m(\sigma)}$ and $d^{m(\sigma)} = 1 - d^\sigma$).

$$\widehat{ATT}_x^{bi} = \frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma \left(\widehat{\mathcal{Y}}_{|1,x^\sigma} - \widehat{\mathcal{Y}}_{|0,x^{m(\sigma)}} \right) \quad (56.107)$$

$$+ \frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma (\epsilon^\sigma - \epsilon^{m(\sigma)}) \quad (56.108)$$

$$\widehat{ATT}_x^{bi} = \underbrace{\frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma \left(\widehat{\mathcal{Y}}_{|1,x^\sigma} - \widehat{\mathcal{Y}}_{|0,x^\sigma} \right)}_{ATT_x^{unbi}} \quad (56.109)$$

$$+ \underbrace{\frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma \left(\widehat{\mathcal{Y}}_{|0,x^\sigma} - \widehat{\mathcal{Y}}_{|0,x^{m(\sigma)}} \right)}_{\Delta ATT_x} \quad (56.110)$$

$$+ \underbrace{\frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma (\epsilon^\sigma - \epsilon^{m(\sigma)})}_{\mathcal{E}_x} \quad (56.111)$$

$$ATT_x^{unbi} = \underbrace{\widehat{ATT}_x^{bi} - \Delta ATT_x}_{\widehat{ATT}_x^{unbi}} - \mathcal{E}_x \quad (56.112)$$

By the Central Limit Theorem, for large N_x , this sum over $\sigma \in A_x$ of i.i.d. summands is normally distributed

$$\sqrt{N_x} ATT_x^{unbi} \sim \mathcal{N}(x; 0, var) \quad (56.113)$$

The reason for the $\sqrt{N_x}$ normalization is that we want the variance to be proportional to N_x .

$$var = N_x \langle ATT_x^{unbi}, ATT_x^{unbi} \rangle \quad (56.114)$$

$$= N_x \langle \widehat{ATT}_x^{unbi}, \widehat{ATT}_x^{unbi} \rangle + N_x \langle \mathcal{E}_x, \mathcal{E}_x \rangle \quad (56.115)$$

QED

56.15 Propensities

It is often the case that the discrete vector \underline{x}^σ has too many possible values to make matching possible. In such cases, it is convenient to map the space of vectors \underline{x}^σ to the real line. One very convenient choice for that map is the **propensity score**, which is defined as

$$g(x^\sigma) = P(\underline{d}^\sigma = 1 | \underline{x}^\sigma). \quad (56.116)$$

$P(\underline{d}^\sigma = 1 | \underline{x}^\sigma)$ is easy to calculate from the dataset. $g(x) = N_{1,x}/N_x$.

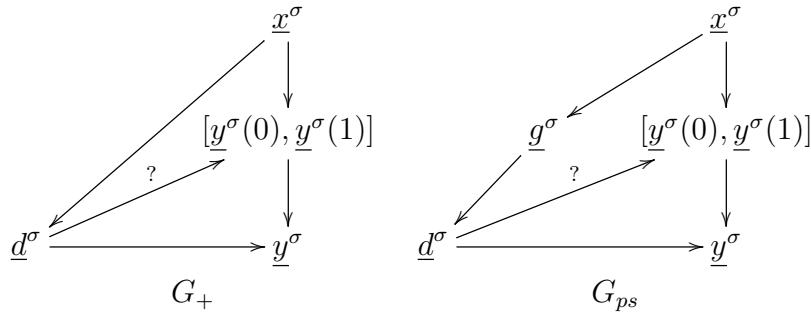


Figure 56.14: Bnet G_{ps} used when doing propensity scoring.

To use the propensity score, one replaces the bnet G_+ by the bnet G_{ps} as shown in Fig.56.14. The TPMs, printed in blue, for the 2 nodes of G_{ps} that differ from the nodes of G_+ , are as follows:

$$P(g^\sigma | x^\sigma) = \delta(g^\sigma, g(x^\sigma)) \quad (56.117)$$

$$P(d^\sigma | g^\sigma) = g^\sigma d^\sigma + (1 - g^\sigma)(1 - d^\sigma) \quad (56.118)$$

Note that these TPMs are self-consistent because

$$P(d|x) = \sum_g P(d|g)P(g|x) \quad (56.119)$$

$$= g(x)d + [1 - g(x)](1 - d) \quad (56.120)$$

$$= P(\underline{d} = 1|x)d + [1 - P(\underline{d} = 1|x)](1 - d) \quad (56.121)$$

$$= P(d|x) \quad (56.122)$$

We would like to do **propensity score strata-matching** by matching g-strata instead of x-strata. PO calculations for x- strata-matching use the TPMs for $P(d|x)$, $P(x)$ and $P(y|d, x)$. To do g- strata-matching using the same equations, but with x replaced by g , we would need to solve for $P(d|g)$, $P(g)$ and $P(y|d, g)$ in terms of $P(d|x)$, $P(x)$ and $P(y|d, x)$. We solve for those next.

From the TPMs for G_{ps} , one has

$$\boxed{P(d|g) = gd + (1 - g)(1 - d)} \quad (56.123)$$

and

$$\boxed{P(g) = \sum_x \overbrace{\delta(g, g(x))}^{P(g|x)} P(x)} \quad (56.124)$$

Next, note that

$$P(y|d, g) = \sum_x P(y|d, x)P(x|g) \quad (56.125)$$

so we need to find $P(x|g)$. Since

$$P(x|g) = \frac{P(g|x)P(x)}{P(g)} \quad (56.126)$$

$$= \frac{\delta(g, g(x))P(x)}{P(g)} \quad (56.127)$$

we finally get

$$\boxed{P(y|d, g) = \sum_x P(y|d, x) \frac{\delta(g, g(x))P(x)}{P(g)}} \quad (56.128)$$

Eq.(56.128) looks complicated, but all it is saying is that

$$P(y|d, g)P(g) = \sum_{x \in g^{-1}(g)} P(y|d, x)P(x), \quad (56.129)$$

where $g^{-1}(g) = \{x : g(x) = g\}$. In general,

$$\sum_x \delta(g, g(x)) = \sum_{x \in g^{-1}(g)} \quad (56.130)$$

Recall that for any treatment effect $\mathcal{E} \in \{ATE, ATT, ATU, SB, SDO\}$, we can estimate \mathcal{E}_x , and then calculate \mathcal{E} from it, using

$$\mathcal{E} = \sum_x P(x)\mathcal{E}_x. \quad (56.131)$$

Hence

$$\mathcal{E} = \sum_g \sum_x \delta(g, g(x))P(x)\mathcal{E}_x \quad (56.132)$$

$$= \sum_g \sum_{x \in g^{-1}(g)} P(x)\mathcal{E}_x \quad (56.133)$$

Let

$$g_{d|x} = g_d(x) = P(d|x) \quad (56.134)$$

for $d \in \{0, 1\}$. Note that $g_{0|x} + g_{1|x} = 1$. We will refer to $g_d(x)$ for $d \in \{0, 1\}$ as a two **propensities** and to $g_1(x)$ as the **propensity score**.

Define

$$\delta_{y|x} = P(y|d = 1, x) - P(y|d = 0, x) \quad (56.135)$$

Note that

$$\delta_{y|x} = \frac{P(y, d = 1|x)}{g_{1|x}} - \frac{P(y, d = 0|x)}{g_{0|x}} \quad (56.136)$$

$$= \frac{P(y, d = 1|x)g_{0|x} - P(y, d = 0|x)g_{1|x}}{g_{0|x}g_{1|x}} \quad (56.137)$$

$$= \frac{P(y, d = 1|x) - P(y|x)g(x)}{g(x)(1 - g(x))} \quad (56.138)$$

Dividing each term in a sum over $d \in \{0, 1\}$ by $g_d(x)$ is often called **inverse probability (or propensity) weighting** (IPW). $g_d(x) = P(\underline{d} = d|x)$ is the likelihood of strata x , so dividing each term by this likelihood increases the contribution to the sum by less likely strata and decreases the contribution by more likely strata.

Note that the backdoor adjustment formula can be expressed as an IPW:

$$P(y|\mathcal{D}\underline{d} = d) = \sum_x P(y|d, x)P(x) \quad (56.139)$$

$$= \sum_x \frac{P(y, d, x)}{P(d|x)} \quad (56.140)$$

Note that $ATE = ACE$ can be expressed in terms of propensities:

$$ACE = \sum_y y[P(y|\mathcal{D}\underline{d} = 1) - P(y|\mathcal{D}\underline{d} = 0)] \quad (56.141)$$

$$= \sum_x P(x) \sum_y y [P(y|d = 1, x) - P(y|d = 0, x)] \quad (56.142)$$

$$= \sum_x P(x) \sum_y \underbrace{y\delta_{y|x}}_{ACE_x} \quad (56.143)$$

Note that

$$P(y(c)|d) = \sum_x P(y(c)|d, x)P(x|d) \quad (56.144)$$

$$= \sum_x P(y(c)|c, x)P(x|d) \text{ (by CIA)} \quad (56.145)$$

$$= \sum_x P(y|c, x)P(x|d) \text{ (by SUTVA)} \quad (56.146)$$

It's instructive to express all the other treatment effects besides ATE in terms of propensities:

$$ATT = \sum_y y[P(\underline{y}(1) = y|d = 1) - P(\underline{y}(0) = y|d = 1)] \quad (56.147)$$

$$= \sum_x P(x|d = 1) \sum_y y [P(y|\underline{d} = 1, x) - P(y|\underline{d} = 0, x)] \text{ (by Eq.(56.146))} \quad (56.148)$$

$$= \sum_x P(x|d = 1) \sum_y y \delta_{y|x} \quad (56.149)$$

$$= \sum_x P(x) \underbrace{\frac{1}{P(\underline{d} = 1)} \sum_y y g_{1|x} \delta_{y|x}}_{ATT_x} \quad (56.150)$$

$$ATU = \sum_x P(x) \underbrace{\frac{1}{P(\underline{d} = 0)} \sum_y y g_{0|x} \delta_{y|x}}_{ATU_x} \quad (56.151)$$

$$SB = \sum_y y [P(\underline{y}(0) = y | d = 1) - P(\underline{y}(0) = y | d = 0)] \quad (56.152)$$

$$= \sum_y y \sum_x P(y | \underline{d} = 0, x) [P(x | d = 1) - P(x | d = 0)] \text{ (by Eq.(56.146))} \quad (56.153)$$

$$= \sum_x P(x) \underbrace{\sum_y y \frac{P(y, \underline{d} = 0 | x)}{g_{0|x}} \left[\frac{g_{1|x}}{P(\underline{d} = 1)} - \frac{g_{0|x}}{P(\underline{d} = 0)} \right]}_{SB_x} \quad (56.154)$$

$$SDO = \sum_y y [P(\underline{y}(1) = y | d = 1) - P(\underline{y}(0) = y | d = 0)] \quad (56.155)$$

$$= \sum_x P(x) \underbrace{\sum_y y \left[\frac{P(y, \underline{d} = 1 | x)}{P(\underline{d} = 1)} - \frac{P(y, \underline{d} = 0 | x)}{P(\underline{d} = 0)} \right]}_{SDO_x} \text{ (by SUTVA)} \quad (56.156)$$

56.16 Propensity based estimators of treatment effects

In the strata-matching section 56.14, we gave an estimate of ACE_x . In strata-matching, one fills in the missing counterfactual values via the map $m()$. This is justified because, by CIA (weak-d limit), the counterfactual distributions are assumed to equal the factual distributions (see Fig. 56.6). In this section, we give estimators of treatment effects that are based on the formulae derived in Section 56.15. The estimators given in this section do no require the map $m()$, because the identification of counterfactual distributions with factual ones was used to derive the formulae of Section 56.15.

Eq.(56.143) suggests the estimator

$$\widehat{ACE}_x = \frac{1}{N_x} \sum_{\sigma \in A_x} y^\sigma \left[\frac{d^\sigma - g(x^\sigma)}{g(x^\sigma)(1 - g(x^\sigma))} \right] \quad (56.157)$$

$$= \frac{1}{N_x} \sum_{\sigma \in A_x} \left[\frac{d^\sigma y^\sigma}{g_{1|x^\sigma}} - \frac{(1 - d^\sigma)y^\sigma}{g_{0|x^\sigma}} \right] \quad (56.158)$$

This estimator is unbiased (because it doesn't have a source of noise like the strata-matching estimators do), but it is still possible to improve it. We next define another

ACE estimator called **Doubly Robust Estimator** (DRE) that is also unbiased and has smaller variance.

Let $\widehat{\mathcal{Y}}_{|d,x}$ be an estimate of $\mathcal{Y}_{|d,x} = E_{y|d,x}[y]$ that is obtained, for example, via Linear Regression. Define the DRE estimator

$$\widehat{ACE}_x^{DRE} = \widehat{\mathcal{Y}}_{1|x}^{DRE} - \widehat{\mathcal{Y}}_{0|x}^{DRE} \quad (56.159)$$

where

$$\widehat{\mathcal{Y}}_{1|x}^{DRE} = \frac{1}{N_x} \sum_{\sigma \in A_x} \left[\frac{d^\sigma(y^\sigma - \widehat{\mathcal{Y}}_{|1,x^\sigma})}{g_{1|x^\sigma}} + \widehat{\mathcal{Y}}_{|1,x^\sigma} \right] \quad (56.160)$$

and

$$\widehat{\mathcal{Y}}_{0|x}^{DRE} = \frac{1}{N_x} \sum_{\sigma \in A_x} \left[\frac{(1 - d^\sigma)(y^\sigma - \widehat{\mathcal{Y}}_{|0,x^\sigma})}{g_{0|x^\sigma}} + \widehat{\mathcal{Y}}_{|0,x^\sigma} \right]. \quad (56.161)$$

The DRE estimator $\widehat{\mathcal{Y}}_{1|x}^{DRE}$ requires first estimating 2 preparatory quantities, $\widehat{\mathcal{Y}}_{|1,x^\sigma}$ and $g_{1|x}$. It's called doubly robust because it remains unbiased even if one of the estimates of those 2 preparatory quantities is wrong, but not if both are wrong. Let's check this.

- Suppose the propensity $g_{1|x}$ is slightly wrong. So what because

$$E_{\sigma|1,x} \left[\frac{d^\sigma(y^\sigma - \widehat{\mathcal{Y}}_{|1,x})}{g_{1|x}} \right] = 0 \quad (56.162)$$

- Suppose $\widehat{\mathcal{Y}}_{|1,x}$ is slightly wrong. So what because

$$E_{\sigma|1,x} \left[\frac{-d^\sigma \widehat{\mathcal{Y}}_{|1,x} + g_{1|x} \widehat{\mathcal{Y}}_{|1,x}}{g_{1|x}} \right] = 0 \quad (56.163)$$

A similar argument can be used to show that $\widehat{\mathcal{Y}}_{0|x}^{DRE}$ is doubly robust too.

56.17 Positivity

Positivity or **non-zero overlap** is defined as the requirement that for all layers x ,

$$0 < \underbrace{P(d^\sigma = 1 | \underline{x}^\sigma = x)}_{g_{1|x}} < 1 \quad (56.164)$$

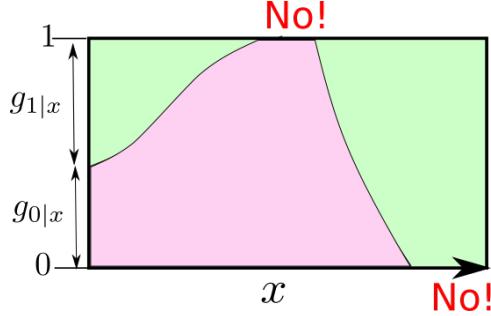


Figure 56.15: Pictorial representation of positivity. $g_{0|x} + g_{1|x} = 1$. $g_{0|x} = 0$ and $g_{1|x} = 0$ are forbidden.

or, equivalently,

$$\underbrace{P(\underline{d}^\sigma = 1 | \underline{x}^\sigma = x)}_{g_{1|x}} > 0 \quad \text{and} \quad \underbrace{P(\underline{d}^\sigma = 0 | \underline{x}^\sigma = x)}_{g_{0|x}} > 0. \quad (56.165)$$

In other words, for each layer x , there is a non-zero probability of being both treated and untreated. See Fig.56.15 for a pictorial representation of positivity.

If positivity is violated for some layer x , then

- the propensity based estimate Eq.(56.158) for ACE_x (which equals ATE_x) is undefined.
- all strata-matching estimates of \mathcal{E}_x that use the matching function $m()$ are undefined because that function is undefined if $A_{0,x} = \emptyset$ or $A_{1,x} = \emptyset$.

If a quantity (estimand) can be estimated, it is said to be **do-identifiable** (i.e., expressible without do() operators). If positivity is violated, then $ACE = ATE$ is not identifiable.

When $P(d|x)$ becomes 0 or 1 for some x , the arrow $\underline{x} \rightarrow \underline{d}$ becomes deterministic for that x . This situation is the very antithesis of RCTs, wherein the influence exerted by \underline{x}^σ on \underline{d}^σ is uniformly random and therefore ignorable. Hence, it is perhaps not too surprising that a violation of positivity makes $ACE = ATE$ not identifiable.

56.18 Multi-time PO bnets (Panel Data)

In this section, we will discuss Multi-time PO bnets (MT-PO).

A **time-series** is a function $f : D \rightarrow \mathbb{R}$ whose domain D is a discrete set of times. A time-series usually describes a single unit σ (i.e., an individual) in a population.

An **observational study** (or **analysis** or **model**) can be cross-sectional or longitudinal. A **cross-sectional study** collects and analyzes a **cross-sectional**

dataset; i.e., a dataset for a population at a single time. A **longitudinal study or panel study** collects and analyzes a **longitudinal dataset**; i.e., a dataset for a population at multiple times. Thus, a longitudinal study consists of one or more time-series.

Let $\mathcal{T} = \{t_0, t_1, \dots, t_{nt-1}\}$. For any time-series $a_t : \mathcal{T} \rightarrow \mathbb{R}$, define

$$E_t a_t = \frac{1}{nt} \sum_{t \in \mathcal{T}} a_t \quad (56.166)$$

$$\Delta_t a_t = a_t - E_t a_t \quad (56.167)$$

$$\langle a_t, b_t \rangle_t = E_t \Delta_t a_t \Delta_t b_t \quad (56.168)$$

Consider a quantity a_t^σ that is a function of the time t and of the particular unit σ in a population. a_t^σ is said to be a **t-constant effect** if it is t -independent. a_t^σ is said to be a **homogeneous effect** (antonym: **heterogeneous effect**) if it is σ -independent. Henceforth, we will avoid using the word “effect” for these because that word has already been used for “treatment effect” in PO theory. Instead, we will use the word “quantity”.

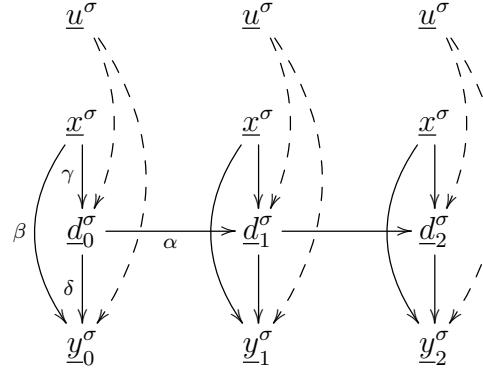


Figure 56.16: Example of multi-time PO bnet with t-constant quantities x^σ, u^σ . The 3 nodes x^σ should be identified as a single node. Likewise, the 3 nodes u^σ should be identified as a single node.

Fig.56.16 gives an example of a multi-time PO bnet (MT-PO). Note that in this example, x^σ and u^σ are t-constant quantities. u^σ is an unobserved confounder and x^σ is an observed confounder. For convenience and simplicity, we will assume linear deterministic TPMs. The TPMs, printed in blue, for the bnet Fig.56.16, are as follows:

$$P(x^\sigma) = P_{\underline{x}}(x^\sigma) \quad (56.169)$$

$$P(u^\sigma) = P_{\underline{u}}(u^\sigma) \quad (56.170)$$

$$P(y_t^\sigma | d_t^\sigma, x^\sigma, u^\sigma) = \mathbb{1}(y_t^\sigma = \delta d_t^\sigma + \beta x^\sigma + u^\sigma) \quad (56.171)$$

$$P(d_{t+1}^\sigma | d_t^\sigma, x^\sigma, u^\sigma) = \mathbb{1}(d_{t+1}^\sigma = \alpha d_t^\sigma + \gamma x^\sigma + u^\sigma) \quad (56.172)$$

Taking time averages of the treatment decision and treatment outcome, we get

$$E_t y_t^\sigma = \delta E_t d_t^\sigma + \beta \underline{x}^\sigma + \underline{u}^\sigma, \quad (56.173)$$

$$E_t d_{t+1}^\sigma = \alpha E_t d_t^\sigma + \gamma \underline{x}^\sigma + \underline{u}^\sigma. \quad (56.174)$$

Subtracting the time averages from the quantities being averaged, we get

$$\Delta_t y_t^\sigma = \delta \Delta_t d_t^\sigma, \quad (56.175)$$

$$\Delta_t d_{t+1}^\sigma = \alpha \Delta_t d_t^\sigma. \quad (56.176)$$

This allows us to find estimators for δ and α :

$$E_\sigma \langle \underline{y}_t^\sigma, \underline{y}_t^\sigma \rangle_t = \delta E_\sigma \langle \underline{y}_t^\sigma, \underline{d}_t^\sigma \rangle_t \quad (56.177)$$

$$\delta = \frac{E_\sigma \langle \underline{y}_t^\sigma, \underline{y}_t^\sigma \rangle_t}{E_\sigma \langle \underline{y}_t^\sigma, \underline{d}_t^\sigma \rangle_t} \quad (56.178)$$

$$E_\sigma \langle \underline{d}_{t+1}^\sigma, \underline{d}_{t+1}^\sigma \rangle_t = \alpha E_\sigma \langle \underline{d}_{t+1}^\sigma, \underline{d}_t^\sigma \rangle_t \quad (56.179)$$

$$\alpha = \frac{E_\sigma \langle \underline{d}_{t+1}^\sigma, \underline{d}_{t+1}^\sigma \rangle_t}{E_\sigma \langle \underline{d}_{t+1}^\sigma, \underline{d}_t^\sigma \rangle_t} \quad (56.180)$$

As shown in Fig.56.17, subtraction of time averages from each node removes the confounder nodes from the bnet of Fig.56.16 (However, this assumes that the confounders are t-constant and that the TPMs are linear deterministic, two very strong assumptions).

$$\begin{array}{ccc} \Delta_t \underline{d}_t^\sigma & \xrightarrow{\alpha} & \Delta_t \underline{d}_{t+1}^\sigma \\ \delta \downarrow & & \downarrow \\ \Delta_t \underline{y}_t^\sigma & & \Delta_t \underline{y}_{t+1}^\sigma \end{array}$$

Figure 56.17: time-average-subtracted (TAS) bnet for the bnet of Fig.56.16.

Chapter 57

Program evaluation and review technique (PERT)

This chapter is based on Refs.[55] and [119].

PERT diagrams are used for scheduling a project consisting of a series of interdependent activities and estimating how long it will take to finish the project. PERT diagrams were invented by the NAVY in 1958 to manage a submarine project. Nowadays they are taught in many business and management courses.

A **PERT diagram** is a Directed Acyclic Graph (DAG) with the following properties. (See Fig.57.2 for an example of a PERT diagram). The nodes \underline{E}_i for $i = 1, 2, \dots, ne$ of a PERT diagram are called **events**. The edges $i \rightarrow j$ of a PERT diagram are called **activities**. An event represents the starting (kickoff) date of one or more activities. A PERT diagram has a single root node ($i = 1$, start event) and a single leaf node ($i = ne$, end event).

The PERT diagram user must initially provide a **Duration Times (DT) table** which gives $(DO_{i \rightarrow j}, DP_{i \rightarrow j}, DM_{i \rightarrow j})$ for each activity $i \rightarrow j$, where

$DO_{i \rightarrow j}$ = optimistic duration time of activity $i \rightarrow j$

$DP_{i \rightarrow j}$ = pessimistic duration time of activity $i \rightarrow j$

$DM_{i \rightarrow j}$ = median duration time of activity $i \rightarrow j$

From the DT table, one calculates:

Duration time of activity $i \rightarrow j$

$$D_{i \rightarrow j} = \frac{1}{6}(DO_{i \rightarrow j} + DP_{i \rightarrow j} + 4DM_{i \rightarrow j}) \quad (57.1)$$

Duration Variance of activity $i \rightarrow j$

$$V_{i \rightarrow j} = \left(\frac{DO_{i \rightarrow j} - DP_{i \rightarrow j}}{DM_{i \rightarrow j}} \right)^2 \quad (57.2)$$

Often, it is convenient to define “dummy” edges with $D_{i \rightarrow j} = 0$. That is perfectly fine.

Define:

TES_i = Earliest start time for event i
 TLS_i = Latest start time for event i
 $slack_i = TLS_i - TES_i$ = slack for event i
 $TEF_{i \rightarrow j} = TES_i + D_{i \rightarrow j}$ = Earliest finish time for activity $i \rightarrow j$.
 $TLF_{i \rightarrow j} = TLS_j - D_{i \rightarrow j}$ = Latest finish time for activity $i \rightarrow j$. See footnote below.¹

A **critical path** is a directed path (i.e., a chain of connected arrows, all pointing in the same direction) going from the start to the end node, such that slack equals zero at every node visited. In a DAG, the neighbors of a node is the union of its parent and children nodes. A critical path must also have all other nodes as neighbors; i.e, the union of the neighbors of every node in the path plus the nodes in the path itself, equals all nodes in the graph.

GOAL of PERT analysis: The main goal of PERT analysis is to find, based on the data of the DT table, the interval $[TES_i, TLS_i]$ giving a lower and an upper bound to the starting time of each node i . Another goal is to find a critical path for the PERT diagram (which represents an entire project). By adding the $D_{i \rightarrow j}$ of each edge of the critical path, one can get the mean value of the total duration of the entire project, and by adding the variances of each edge along the critical path, one can get an estimate of the total variance of the total duration. Knowing the mean and variance of the total duration and assuming a Normal distribution, one can predict the probability that the actual duration will deviate by a certain amount from its mean.

To calculate the interval $[TES_i, TLS_i]$, one follows the following two steps.

1. Assume $TES_1 = 0$ and solve

$$TES_i = \max_{a \in pa(i)} (\underbrace{TES_a + D_{a \rightarrow i}}_{TEF_{a \rightarrow i}}) \quad (57.3)$$

for $i \in [2, ne]$. This recursive equation is solved by what is called “forward propagation”, wherein one moves up the list of nodes i in order of increasing i starting at $i = 1$ with $TES_1 = 0$.

2. Assume $TLS_{ne} = TES_{ne}$ and solve

$$TLS_i = \min_{b \in ch(i)} (\underbrace{TLS_b - D_{i \rightarrow b}}_{TLF_{i \rightarrow b}}) \quad (57.4)$$

¹In the popular educational literature, the edge variables $TEF_{i \rightarrow j}$ and $TLF_{i \rightarrow j}$ are sometimes associated with the nodes, but they are clearly edge variables. This makes things confusing. The reason this is done is that some software draws PERT diagrams as trees whereas other software draws them as DAGs. For trees, storing $TEF_{i \rightarrow j}$ and $TLF_{i \rightarrow j}$ in a node makes some sense but not for DAGs. You will notice that giving specific names to the variables $TEF_{i \rightarrow j}$ and $TLF_{i \rightarrow j}$ is unnecessary. It is possible to delete all mention of their names from this chapter without losing any details. I only declare their names in this chapter so as tell the reader what they are in case he/she hears them mentioned and wonders what they are equal to in our notation.

for $i \in [1, ne - 1]$. This recursive equation is solved by what is called “backward propagation”, wherein one moves down the list of nodes i in order of decreasing i starting at $i = ne$ with $TLS_{ne} = TES_{ne}$. TES_{ne} is known from step 1.

Eqs.(57.3) and (57.4) are illustrated in Fig.57.1.

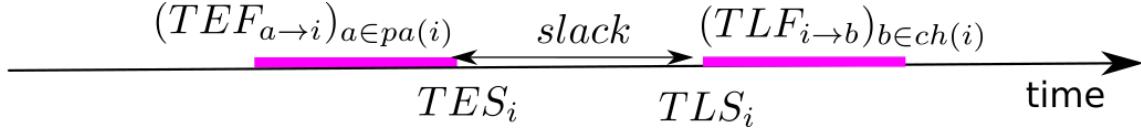


Figure 57.1: TES_i defined from info received from parents of i and TLS_i defined from info received from children of i .

57.1 Example

To illustrate PERT analysis, we end with an example. We present the example in the form of an exercise question and then provide the answer. This example comes from Ref.[55], except for part (e) about bnets, which is our own.

Question: For the PERT diagram of Fig.57.2, calculate the following:

- (a) Interval $[TES_i, TLS_i]$ for all i .
- (b) A critical path for this PERT diagram.
- (c) The mean and variance of the total duration of the critical path.
- (d) The probability that the total duration will be 225 days or less.
- (e) A bnet interpretation of this problem.

Answer to (a) $[TES_i, TLS_i]$ are given by Fig.57.3.

Answer to (b) The critical path is given in red in Fig.57.3. Note that this path does indeed have zero slack at each node it visits and the union of its neighborhood and the path itself encompasses all nodes.

Answer to (c) The mean and variance of the total duration are calculated in Table 57.1.

Answer to (d)

$$P(\underline{x} < 225) = P\left[\frac{\underline{x} - \mu}{\sigma} \leq \frac{225 - 220}{\sqrt{7.73}}\right] \quad (57.5)$$

$$= P[\underline{z} \leq 1.80] \quad (57.6)$$

$$= 0.9641 \quad (57.7)$$

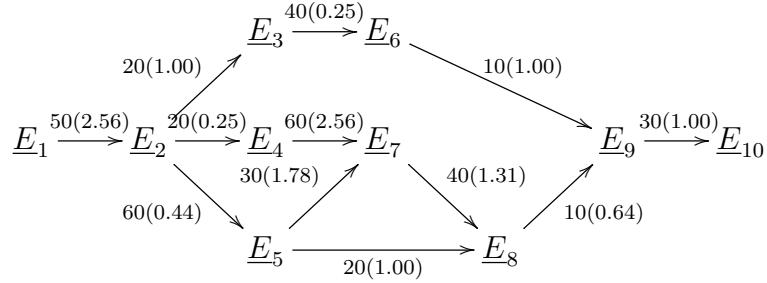


Figure 57.2: Example of a PERT diagram. The numbers attached to the arrows are the duration times $D_{i \rightarrow j}$ in days followed by, enclosed in parentheses, the variance $V_{i \rightarrow j}$ of that duration. The info given in this PERT diagram was derived from a DT table in Ref.[55]. The info in this PERT diagram is sufficient for calculating TES_i and TLS_i for each node i . The results of that calculation are given in Fig.57.3.

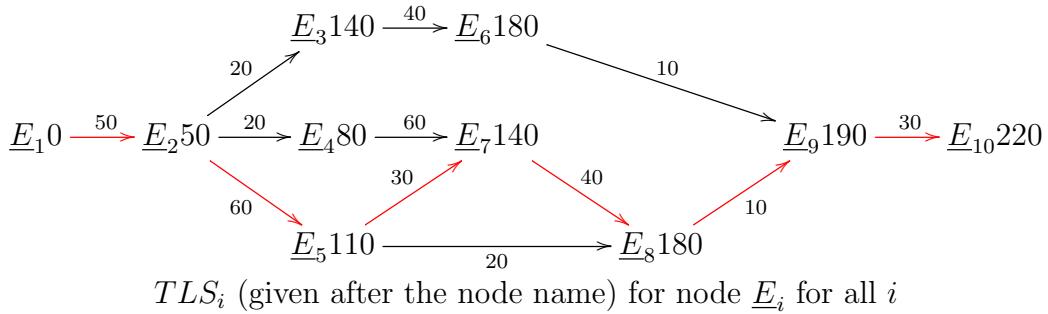
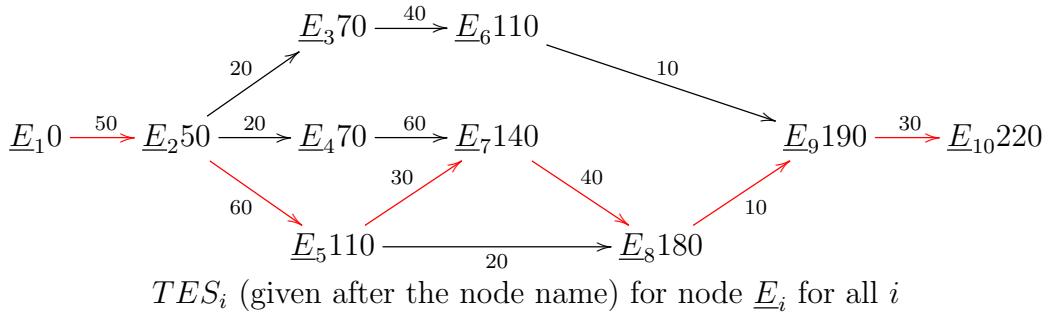


Figure 57.3: Results of calculating TES_i for all i via a forward pass, followed by calculating TLS_i for all i via a backward pass. Critical path indicated in red.

Answer to (e) Define 2 bnets.

1. The first PERT bnet is for calculating TES_i for all i and is given by Fig.57.4.

edge $i \rightarrow j$	duration $D_{i \rightarrow j}$	variance $V_{i \rightarrow j}$
A (1 → 2)	50	2.56
D (2 → 5)	60	0.44
G (5 → 7)	30	1.78
J (7 → 8)	40	1.31
K (8 → 9)	10	0.64
L (9 → 10)	30	1.00
Total	220	7.73

Table 57.1: Calculation of mean and variance of total duration along critical path.

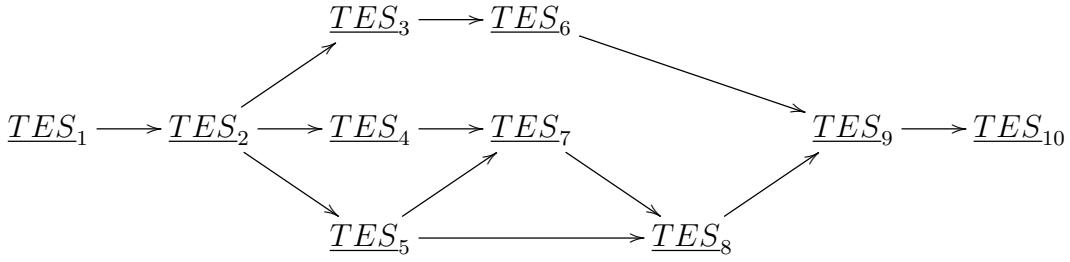


Figure 57.4: bnet for TES_i calculation.

The TPMs, printed in blue, for the bnet Fig.57.4, are as follows (this equation is to be evaluated recursively by a forward pass through the bnet):

$$P(TES_i | (TES_a)_{a \in pa(i)}) = \delta(TES_i, \max_{a \in pa(i)} (TES_a + D_{a \rightarrow i})) \quad (57.8)$$

2. The second PERT bnet is for calculating TLS_i for all i and is given by Fig.57.5. Note that the directions of all the arrows in the PERT diagram Fig.57.2 have been reversed so Fig.57.5 is a time reversed graph.

The TPMs, printed in blue, for the bnet Fig.57.5, are as follows (this equation is to be evaluate recursively by a backward pass through the bnet):

$$P(TLS_i | (TLS_b)_{b \in pa(i)}) = \delta(TLS_i, \min_{b \in pa(i)} (TLS_b - D_{b \rightarrow i}^T)) , \quad (57.9)$$

where $D_{i \rightarrow j}^T = D_{j \rightarrow i}$.

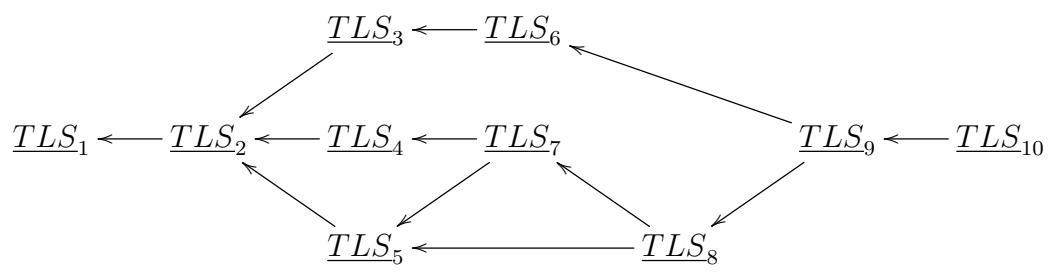


Figure 57.5: bnet for TLS_i calculation.

Chapter 58

Random Forest and Bagging

This chapter is based on Refs.[76] and [120].

Chapter 13 defines decision trees (dtrees) and explains how to construct them. A Random Forest (RF) is an ensemble of dtrees. The RF algorithm is a method of, given a dataset, constructing a RF and averaging over the classifier functions of the RF to produce an ensemble classifier.

The RF algo uses the method of Bootstrap Aggregating (aka Bagging), which is discussed in detail below. Bagging is a method of constructing an ensemble of datasets (called **bootstrap datasets or bags**) that are fairly uncorrelated. Each of these bags is used to train a **bag-classifier**. The bag-classifiers are averaged over to produce an **ensemble classifier, or e-classifier** for short. Bagging can be used to train any type of bag-classifier, but it was invented with dtrees in mind, and is still most commonly used to train dtrees.

Boosting (see Chapter 1 on AdaBoost and Chapter 80 on XGBoost) is another method, besides Bagging, of constructing a classifier function from an ensemble of classifier functions. These two methods are most commonly applied to dtrees: Boosting for an ensemble of small dtrees, and Bagging for a random forest (which is an ensemble of dtrees that are usually much more complicated than small dtrees).

58.1 Bagging (with fully-featured bags)

In this section we discuss the bagging algorithm. As already mentioned, bagging is usually used to train dtrees. In this section, we explain bagging in general, not just for dtrees.

Let $L = [0, 1, 2, \dots, nsam - 1]$ be a list of individuals (samples) in a population. In this chapter, we will use the notation $A^\sigma = A[\sigma]$ and $\vec{A} = [A^\sigma : \sigma \in L]$ for a list (vector, 1-D array) indexed by L . We will refer to $DS = (\vec{x}, \vec{y})$ where $x^\sigma \in S_{\underline{x}}$, $y^\sigma \in S_{\underline{y}}$, as a dataset. If L_j is a list (possibly with duplicate items) such that $set(L_j) \subset set(L)$, then define $DS_j = (\vec{x}, \vec{y})_{L_j} = ((x^\sigma)_{\sigma \in L_j}, (y^\sigma)_{\sigma \in L_j})$. We will refer to DS_j as the **restriction of (\vec{x}, \vec{y}) to L_j** .

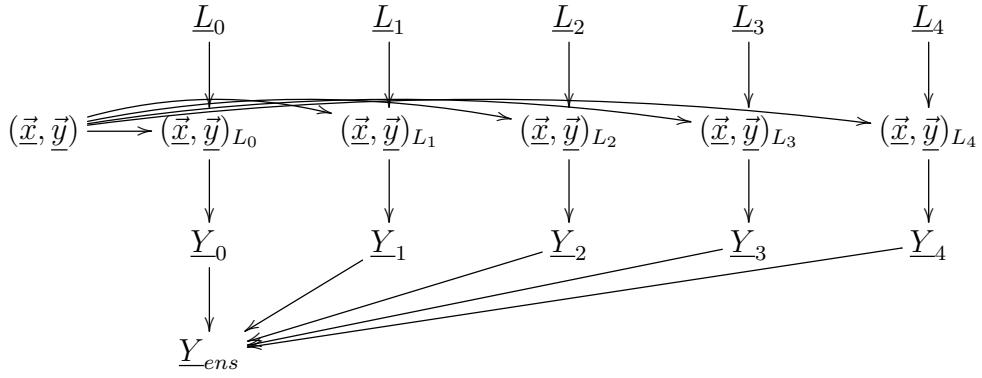


Figure 58.1: Bnet for Random Forest (RF) with 5 bags.

We will refer to a function $Y : S_x \rightarrow S_c$ as a classifier. It maps vector of vector of features $x \in S_x$ to a class $c \in S_c$. Below, Y_b for all b and Y_{ens} are classifiers.

Fig.58.1 is a bnet that encapsulates the RF algo. The TPMs, printed in blue, for the bnet Ref.58.1, are as follows.

Let $b \in \{0, 1, 2, \dots, nbags - 1\} = B$ and $\sigma \in L$. Let $L_b^\sigma \in L$ and

$$P(L_b^\sigma) = 1/nsam \quad (58.1)$$

In other words, each item in list L_b is chosen from the items of list L , uniformly at random with replacements. $|L_b| = |L|$ (same size as original). L_b can have duplicate items and be missing items from L .

$$P((\vec{x}, \vec{y})_{L_b} | (\vec{x}, \vec{y}), L_b) = \mathbb{1}(\text{ } (\vec{x}, \vec{y})_{L_b} = \text{restriction of } (\vec{x}, \vec{y}) \text{ to } L_b \text{ }) \quad (58.2)$$

We will refer to (\vec{x}, \vec{y}) as the **original dataset** and to the $(\vec{x}, \vec{y})_{L_b}$ for $b \in B$ as the **bootstrap datasets** or **bags**.

$$P(Y_b | (\vec{x}, \vec{y})_{L_b}) = \mathbb{1}(\text{ } Y_b(\cdot) = \text{classifier trained on dataset } (\vec{x}, \vec{y})_{L_b}. \text{ }) \quad (58.3)$$

$$P(Y_{ens} | (Y_b)_{b \in B}) = \prod_{\sigma} \mathbb{1}(\text{ } Y_{ens}(x^\sigma) = \text{majority}(\{Y_b(x^\sigma) : b \in B\}) \text{ }) \quad (58.4)$$

The **majority()** function can be replaced by an average $\frac{1}{nbags} \sum_b$ in case the set of classes S_c equals \mathbb{R} rather than a finite set. We will refer to Y_{ens} as the **ensemble classifier (e-classifier)** and to the Y_b as the **bag-classifiers**.

Define (these definitions are illustrated in Fig.58.2)

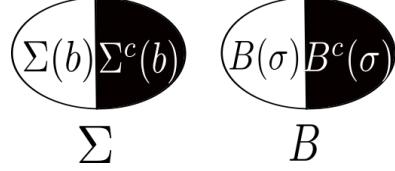


Figure 58.2: $\Sigma(b)$ and $\Sigma^c(b)$ are disjoint sets whose union is Σ . $B(\sigma)$ and $B^c(\sigma)$ are disjoint sets whose union is B .

$$\Sigma = \text{set}(L), \quad \Sigma(b) = \text{set}(L_b), \quad \Sigma^c(b) = \Sigma - \Sigma(b) \quad (58.5)$$

and

$$B(\sigma) = \{b \in B : \sigma \in \Sigma(b)\}, \quad B^c(\sigma) = B - B(\sigma) \quad (58.6)$$

$\Sigma(b)$ is the set of **in-the-b-bag individuals** and $\Sigma^c(b)$ is the set of **out-of-the-b-bag (OOB) individuals**. $B(\sigma)$ is the set of bags that contain individual σ , and $B^c(\sigma)$ is the set of bags that don't.

The **OOB error** is defined as

$$err = \sum_{\sigma \in L} \mathbb{1}(B^c(\sigma) \neq \emptyset) \mathbb{1}(y^\sigma \neq \text{majority}([Y_b(x^\sigma) : b \in B^c(\sigma)])) . \quad (58.7)$$

Empirical results supposedly show that OOB error is comparable in accuracy to the error calculated by doing cross-validation (CV) (see Chapter 11), although CV error is considered more dependable.

58.2 Bagging (with randomly-shortened bags)

Suppose the feature vector x^σ in the dataset $DS = (\vec{x}, \vec{y})$ has nf components; i.e., $x^\sigma = (x_0^\sigma, x_1^\sigma, \dots, x_{nf-1}^\sigma) \in S_{\underline{x}_0} \times S_{\underline{x}_1} \times \dots \times S_{\underline{x}_{nf-1}} = S_{\underline{x}}$.

For each bag DS_b , one chooses at random $nf' = \sqrt{nf}$ out of the nf features, and discards the remaining features from DS_b , thus producing a new, **randomly-shortened-bag (rs-bag)** DS'_b . Each rs-bag is then used to train a bag-classifier, usually a dtree, using the methods for dtree SL described in Chapter 13. Using rs-bags is called the **random subspace method**. The reason for using rs-bags is that they further decorrelate the set of bags used to train bag-classifiers.

Chapter 59

Recurrent Neural Networks

This chapter is mostly based on Ref.[31].

This chapter assumes you are familiar with the material and notation of Chapter 50 on plain Neural Nets.

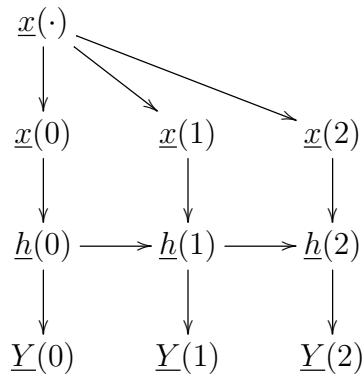


Figure 59.1: Simple example of RNN with $T = 3$

Suppose

T is a positive integer.

$t = 0, 1, \dots, T - 1$,

$\underline{x}_i(t) \in \mathbb{R}$ for $i = 0, 1, \dots, nx - 1$,

$\underline{h}_i(t) \in \mathbb{R}$ for $i = 0, 1, \dots, nh - 1$,

$\underline{Y}_i(t) \in \mathbb{R}$ for $i = 0, 1, \dots, ny - 1$,

$W^{h|x} \in \mathbb{R}^{nh \times nx}$,

$W^{h|h} \in \mathbb{R}^{nh \times nh}$,

$W^{y|h} \in \mathbb{R}^{ny \times nh}$,

$b^y \in \mathbb{R}^{ny}$,

$b^h \in \mathbb{R}^{nh}$.

Henceforth, $x(\cdot)$ will mean the array of $x(t)$ for all t .

The simplest kind of recurrent neural network (RNN) has the bnet Fig.59.1 with arbitrary T . The node TPMs, printed in blue, for this bnet, are as follows.

$$P(x(\cdot)) = \text{given} \quad (59.1)$$

$$P(x(t)) = \delta(x(t), [x(\cdot)]_t) \quad (59.2)$$

$$P(h(t) | h(t-1), x(t)) = \delta(h(t), \mathcal{A}(W^{h|x}x(t) + W^{h|h}h(t-1) + b^h)) , \quad (59.3)$$

where $h(-1) = 0$.

$$P(Y(t) | h(t)) = \delta(Y(t), \mathcal{A}(W^{y|h}h(t) + b^y)) \quad (59.4)$$

Define

$$W^h = [W^{h|x}, W^{h|h}, b^h] , \quad (59.5)$$

and

$$W^y = [W^{y|h}, b^y] . \quad (59.6)$$

The bnet of Fig.59.1 can be used for classification once its parameters W^h and W^y have been optimized. To optimize those parameters via gradient descent, one can use the bnet of Fig.59.2.

Let $\sigma = 0, 1, \dots, nsam(\vec{x}) - 1$ be the labels for a minibatch of samples. Below, we will write $A^\sigma = A[\sigma]$ for the σ component of any vector \vec{A} . The TPMs, printed in blue, for bnet Fig.59.2, are as follows.

$$P(x(\cdot)^\sigma) = \text{given} \quad (59.7)$$

$$P(x(t)^\sigma) = \delta(x(t)^\sigma, [x(\cdot)]_t^\sigma) \quad (59.8)$$

$$P(h(t)^\sigma | h(t-1)^\sigma, x(t)^\sigma) = \delta(h(t)^\sigma, \mathcal{A}(W^{h|x}x(t)^\sigma + W^{h|h}h(t-1)^\sigma + b^h)) \quad (59.9)$$

$$P(Y(t)^\sigma | h(t-1)^\sigma) = \delta(Y(t)^\sigma, \mathcal{A}(W^{y|h}h(t-1)^\sigma + b^y)) \quad (59.10)$$

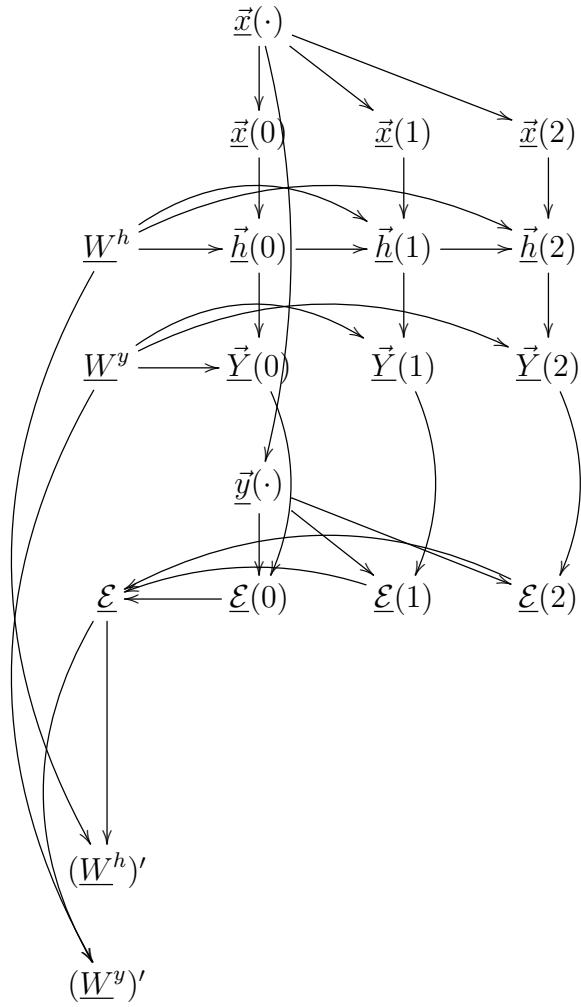


Figure 59.2: RNN bnet used to optimize parameters W^h and W^y of RNN bnet Fig.59.1.

$$P(y(\cdot)^\sigma | x(\cdot)^\sigma) = \text{given} \quad (59.11)$$

$$P(\mathcal{E}(t) | \vec{y}(t), \vec{Y}(t)) = \frac{1}{nsam(\vec{x})} \sum_{\sigma} d(y(t)^\sigma, Y(t)^\sigma) , \quad (59.12)$$

where

$$d(y, Y) = |y - Y|^2 . \quad (59.13)$$

If $y, Y \in [0, 1]$, one can use this instead

$$d(y, Y) = XE(y \rightarrow Y) = -y \ln Y - (1 - y) \ln(1 - Y). \quad (59.14)$$

$$P(\mathcal{E} | \mathcal{E}(\cdot)) = \delta(\mathcal{E}, \sum_t \mathcal{E}(t)) \quad (59.15)$$

For $a = h, y$,

$$P(W^a) = \text{given}. \quad (59.16)$$

The first time it is used, W^a is arbitrary. Afterwards, it is determined by previous horizontal stage.

$$P((W^a)' | \mathcal{E}, W^a) = \delta((W^a)', W^a - \eta^a \partial_{W^a} \mathcal{E}). \quad (59.17)$$

$\eta^a > 0$ is the learning rate for W^a .

59.1 Language Sequence Modeling

Estimate $P(x(\cdot))$ empirically. We can use this to:

- predict the probability of a sentence,
example: Get $P(x(0), x(1), x(2))$.
- predict the most likely next word in a sentence,
example: Get $P(x(2)|x(0), x(1))$.
- generate fake sentences.
example:
Get $x(0) \sim P(x(0))$.
Next get $x(1) \sim P(x(1)|x(0))$.
Next get $x(2) \sim P(x(2)|x(0), x(1))$.

59.2 Other types of RNN

Let $\mathcal{T} = \{0, 1, \dots, T - 1\}$, and $\mathcal{T}^x, \mathcal{T}^y \subset \mathcal{T}$. Above, we assumed that $\underline{x}(t)$ and $\underline{Y}(t)$ were both defined for all $t \in \mathcal{T}$. More generally, they might be defined only for subsets of \mathcal{T} : $\underline{x}(t)$ for $t \in \mathcal{T}^x$ and $\underline{Y}(t)$ for $t \in \mathcal{T}^y$. If $|\mathcal{T}^x| = 1$ and $|\mathcal{T}^y| > 1$, we say the RNN bnet is of the **1 to many** kind. In general, can have **1 to 1**, **1 to many**, **many to 1**, **many to many** RNN bnets.

Plain RNNs can suffer from the **vanishing or exploding gradients problem**. There are various ways to mitigate this (e.g., good choice of initial W^h and

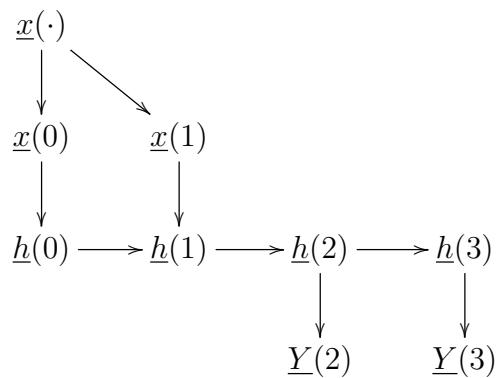


Figure 59.3: RNN bnet of the many to many kind. This one can be used for translation. $x(0)$ and $x(1)$ might denote two words of an English sentence, and $Y(2)$ and $Y(3)$ might be their Italian translation.

W^y , good choice of activation functions, regularization). Or by using GRU or LSTM (discussed below). **GRU and LSTM** were designed to mitigate the vanishing or exploding gradients problem. They are very popular in NLP (Natural Language Processing).

59.2.1 Long Short Term Memory (LSTM) unit (1997)

This section is based on Wikipedia article Ref.[107]. In this section, \odot will denote the Hadamard matrix product (elementwise product).

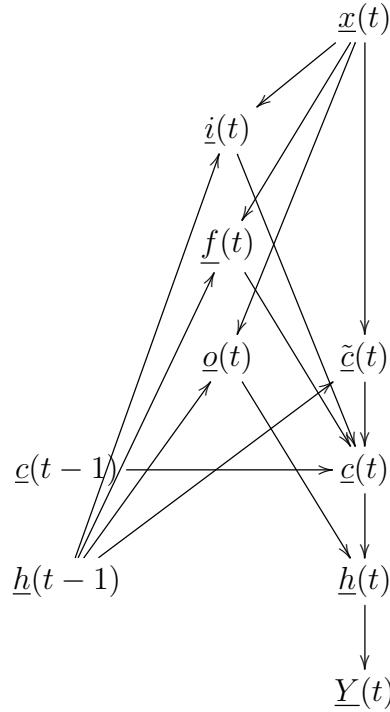


Figure 59.4: bnet for a Long Short Term Memory (LSTM) unit.

Let

$\underline{x}(t) \in \mathbb{R}^{nx}$: **input state vector** to the LSTM unit

$\underline{f}(t) \in \mathbb{R}^{nh}$: **forget activation** vector

$\underline{i}(t) \in \mathbb{R}^{nh}$: **input activation** vector

$\underline{o}(t) \in \mathbb{R}^{nh}$: **output activation** vector

$\underline{h}(t) \in \mathbb{R}^{nh}$: **hidden state vector**

$\underline{\tilde{c}}(t) \in \mathbb{R}^{nh}$: **cell activation** vector

$\underline{c}(t) \in \mathbb{R}^{nh}$: **cell state vector**

$\underline{Y}(t) \in \mathbb{R}^{ny}$: **classification** of $\underline{x}(t)$.

$W \in \mathbb{R}^{nh \times nx}$, $U \in \mathbb{R}^{nh \times nh}$ and $b \in \mathbb{R}^{nh}$: weight matrices and bias vectors, parameters learned by training.

$\mathcal{W}^{y|h} \in \mathbb{R}^{ny \times nh}$: weight matrix

Fig.59.4 is a bnet for a LSTM unit. The TPMs, printed in blue, for this bnet, are as follows.

$$P(f(t)|x(t), h(t-1)) = \mathbb{1}(\quad f(t) = \text{smoid}(W^{f|x}x(t) + U^{f|h}h(t-1) + b^f) \quad) , \quad (59.18)$$

where $h(-1) = 0$.

$$P(i(t)|x(t), h(t-1)) = \mathbb{1}(\quad i(t) = \text{smoid}(W^{i|x}x(t) + U^{i|h}h(t-1) + b^i) \quad) \quad (59.19)$$

$$P(o(t)|x(t), h(t-1)) = \mathbb{1}(\quad o(t) = \text{smoid}(W^{o|x}x(t) + U^{o|h}h(t-1) + b^o) \quad) \quad (59.20)$$

$$P(\tilde{c}(t)|x(t), h(t-1)) = \mathbb{1}(\quad \tilde{c}(t) = \tanh(W^{c|x}x(t) + U^{c|h}h(t-1) + b^c) \quad) \quad (59.21)$$

$$P(c(t)|f(t), c(t-1), i(t), \tilde{c}(t)) = \mathbb{1}(\quad c(t) = f(t) \odot c(t-1) + i(t) \odot \tilde{c}(t) \quad) \quad (59.22)$$

$$P(h(t)|o(t), c(t)) = \mathbb{1}(\quad h(t) = o(t) \odot \tanh(c(t)) \quad) \quad (59.23)$$

$$P(Y(t)|h(t)) = \mathbb{1}(\quad Y(t) = \mathcal{A}(\mathcal{W}^{y|h}h(t) + b^y) \quad) \quad (59.24)$$

59.2.2 Gated Recurrence Unit (GRU) (2014)

This section is based on Wikipedia article Ref.[90]. In this section, \odot will denote the Hadamard matrix product (elementwise product).

GRU is a more recent (17 years later) attempt at simplifying LSTM.

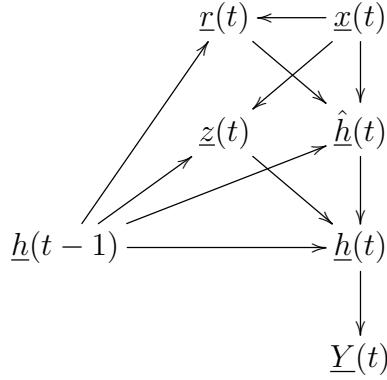


Figure 59.5: bnet for a Gated Recurrent Unit (GRU).

Let

$x(t) \in \mathbb{R}^{nx}$: **input state vector**

$h(t) \in \mathbb{R}^{nh}$: **hidden state vector**

$\hat{h}(t) \in \mathbb{R}^{nh}$: **hidden activation vector**

$z(t) \in \mathbb{R}^{nh}$: **update activation vector**

$r(t) \in \mathbb{R}^{nh}$: **reset activation vector**

$Y(t) \in \mathbb{R}^{ny}$: **classification** of $x(t)$.

$W \in \mathbb{R}^{nh \times nx}$, $U \in \mathbb{R}^{nh \times nh}$ and $b \in \mathbb{R}^{nh}$: weight matrices and bias vectors, parameters learned by training.

$\mathcal{W}^{y|h} \in \mathbb{R}^{ny \times nh}$: weight matrix

Fig.59.5 is a bnet for a GRU. The TPMs, printed in blue, for this bnet, are as follows.

$$P(z(t)|x(t), h(t-1)) = \mathbb{1}(\quad z(t) = \text{smoid}(W^{z|x}x(t) + U^{z|h}h(t-1) + b^z) \quad), \quad (59.25)$$

where $h(-1) = 0$.

$$P(r(t)|x(t), h(t-1)) = \mathbb{1}(\quad r(t) = \text{smoid}(W^{r|x}x(t) + U^{r|h}h(t-1) + b^r) \quad) \quad (59.26)$$

$$P(\hat{h}(t)|x(t), r(t), h(t-1)) = \mathbb{1}(\quad \hat{h}(t) = \tanh(W^{h|x}x(t) + U^{h|h}(r(t) \odot h(t-1)) + b^h) \quad) \quad (59.27)$$

$$P(h(t)|z(t), h(t-1), \hat{h}(t)) = \mathbb{1}(\quad h(t) = (1 - z(t)) \odot h(t-1) + z(t) \odot \hat{h}(t) \quad) \quad (59.28)$$

$$P(Y(t)|h(t)) = \mathbb{1}(\quad Y(t) = \mathcal{A}(\mathcal{W}^{y|h}h(t) + b^y) \quad) \quad (59.29)$$

Chapter 60

Regression Discontinuity Design

This chapter is based on Ref.[6].

This chapter assumes that the reader has read Chapter 56 on Potential Outcomes (PO).

In Regression Discontinuity Design (RDD), one switches the treatment dose \underline{d} from 0 when $\underline{x} < \xi$ to 1 where $\underline{x} > \xi$, where \underline{x} is an observed confounder (call it the **switch confounder**) and ξ is a threshold value for \underline{x} . One measures the jump δ in the treatment outcome \underline{y} as \underline{x} passes through $\underline{x} = \xi$. Then one makes the very reasonable assumption that δ equals¹ $\mathcal{Y}_{1|x=\xi} - \mathcal{Y}_{0|x=\xi} = ATE|_{x=\xi}$ for an imaginary experiment in which the confounder \underline{x} acts as a normal confounder that doesn't switch the treatment dose \underline{d} .

For example, d^σ might be whether an individual is admitted to Harvard Univ., y^σ might be how much money the individual earns for the first 20 years after graduating from Harvard, and x^σ might be his SAT scores. We assume Harvard only admits students with an SAT score higher than ξ .

60.1 PO analysis

The TPMs, printed in blue, for the bnet G_{disc} shown in Fig.60.1, are as follows. Note that the TPMs for the bnet G_{disc} are defined in terms of the TPMs for the bnet G .

$$P(x^\sigma) = \delta(x^\sigma, x) \quad (60.1)$$

$$P(d^\sigma|x^\sigma = x) = \begin{cases} \delta(d^\sigma, 1) & \text{for } x > \xi \\ \delta(d^\sigma, 0) & \text{for } x < \xi \end{cases} \quad (60.2)$$

$$P(y^\sigma|y^\sigma(0), y^\sigma(1), d^\sigma) = \mathbb{1}(y^\sigma = y^\sigma(d^\sigma)) \quad (60.3)$$

¹ATE, which stands for “average treatment effect”, is defined in Chapter 56.

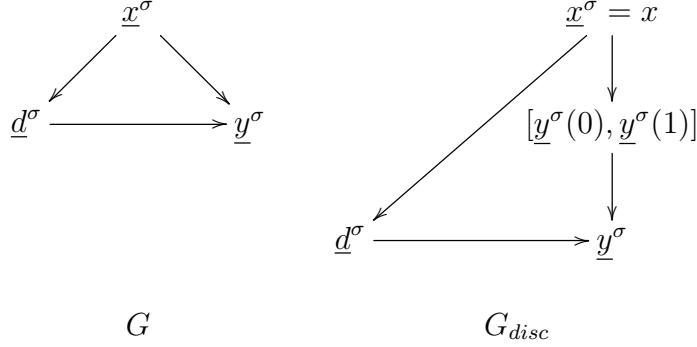


Figure 60.1: 2 bnets used in the PO analysis of RDD. The TPMs for G_{disc} are defined in terms of the TPMs for G . The TPM $P(d^{\sigma}|x^{\sigma})$ for G_{disc} is discontinuous in x^{σ} .

$$P(y^{\sigma}(c)|x) = P(y^{\sigma}(c)|d^{\sigma}, x) = \text{given} \quad (60.4)$$

Define

$$E_{\sigma|x}[y^{\sigma}(c)] = E_{y(c)|x}[\underline{y}(c)] = \mathcal{Y}_{c|x} \quad (60.5)$$

and

$$\xi^{\pm} = \xi \pm \epsilon \quad (60.6)$$

for some infinitesimal $\epsilon > 0$.

See Fig.60.2. In RDD, we assume that if we define the following 2 δ 's, one for bnet G and the other for bnet G_{disc} , then the two δ 's are equal, and they equal a conditional ATE.

$$\delta_{G_{disc}} = \mathcal{Y}_{1|x=\xi+} - \mathcal{Y}_{0|x=\xi-} \quad (60.7)$$

$$\delta_G = \mathcal{Y}_{1|x=\xi} - \mathcal{Y}_{0|x=\xi} \quad (60.8)$$

$$\delta_G = \delta_{G_{disc}} = \delta \quad (60.9)$$

$$\delta = ATE|_{x=\xi} \quad (60.10)$$

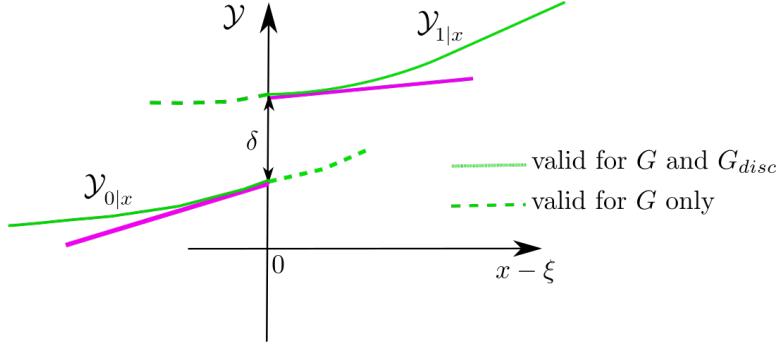


Figure 60.2: The jump δ between $\mathcal{Y}_{1|x}$ and $\mathcal{Y}_{0|x}$ is the same for G and G_{disc} .

60.2 Linear Regression

In this section, we show how to apply linear regression (LR) to the PO analysis of RDD.

y^σ can be fitted as a function of $x \in \mathbb{R}$, for $c^\sigma \in \{0, 1\}$, as follows. Here ϵ^σ is the residual for individual σ and $b_0, m_0, b_1, m_1 \in \mathbb{R}$ are the fit parameters.

$$y^\sigma = [b_0 + m_0(x - \xi)](1 - c^\sigma) + [b_1 + m_1(x - \xi)]c^\sigma + \epsilon^\sigma. \quad (60.11)$$

Note that Eq.(60.11) yields a straight line in the $y^\sigma - x$ plane for $c^\sigma = 0$, and another straight line for $c^\sigma = 1$. These 2 lines are colored magenta in Fig.60.2. We are using the standard symbols b to denote the y-intercept, and m to denote the slope of a straight line.

Taking the expected value of Eq.(60.11), we get

$$\mathcal{Y}_{c|x} = [b_0 + m_0(x - \xi)](1 - c) + [b_1 + m_1(x - \xi)]c. \quad (60.12)$$

Hence,

$$\mathcal{Y}_{1|x=\xi+} = b_1, \quad \mathcal{Y}_{0|x=\xi-} = b_0 \quad (60.13)$$

$$\delta = b_1 - b_0 \quad (60.14)$$

Chapter 61

Reinforcement Learning (RL)

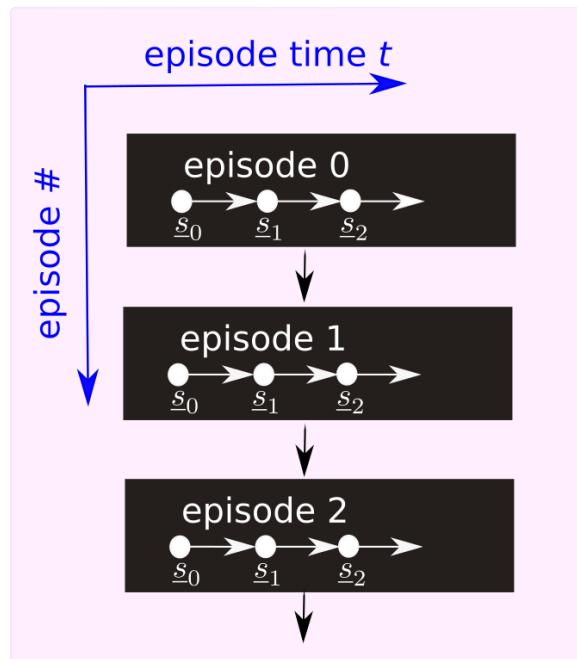


Figure 61.1: Axes for episode time and episode number.

I based this chapter on the following references. Refs.[10][21]

In RL, we consider an “agent” or robot that is learning.

Let $T \in \mathbb{Z}_{>0}$ be the duration time of an **episode** of learning. If $T = \infty$, we say that the episode has an infinite time horizon. A learning episode will evolve towards the right, for times $t = 0, 1, \dots, T - 1$. We will consider multiple learning episodes. The episode number will evolve from top to bottom. This is illustrated in Fig.61.1.

Let $\underline{s}_t \in S_s$ for $t \in \mathbb{Z}_{[0,T-1]}$ be random variables that record the **state** of the agent at various times t .

Let $\underline{a}_t \in S_a$ for $t \in \mathbb{Z}_{[0,T-1]}$ be random variables that record the **action** of the agent at various times t .

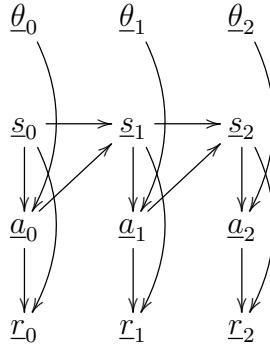


Figure 61.2: State-Action-Reward dynamical bnet

Let $\underline{\theta}_t \in S_{\underline{\theta}}$ for $t \in \mathbb{Z}_{[0,T-1]}$ be random variables that record the **policy parameters** at various times t .

For $\underline{X} \in \{\underline{s}, \underline{a}, \underline{\theta}\}$, define \underline{X} followed by a dot to be the vector

$$\underline{X}_{\cdot} = [\underline{X}_0, \underline{X}_1, \dots, \underline{X}_{T-1}] . \quad (61.1)$$

Also let

$$\underline{X}_{\geq t} = [\underline{X}_t, \underline{X}_{t+1}, \dots, \underline{X}_{T-1}] . \quad (61.2)$$

Fig.61.2 shows the basic State-Action-Reward bnet for an agent that is learning. The TPMs, printed in blue, for the bnet Fig.61.2, are as follows.

$$P(a_t|s_t, \theta_t) = \text{given.} \quad (61.3)$$

$P(a_t|s_t, \theta_t)$ is called a **policy with parameter** θ_t .

$$P(s_t|s_{t-1}, a_{t-1}) = \text{given.} \quad (61.4)$$

$P(s_t|s_{t-1}, a_{t-1})$ is called the **TPM of the model**. $P(s_t|s_{t-1}, a_{t-1})$ reduces to $P(s_0)$ when $t = 0$.

$$P(r_t|s_t, a_t) = \delta(r_t, r(s_t, a_t)) . \quad (61.5)$$

$r : S_{\underline{s}} \times S_{\underline{a}} \rightarrow \mathbb{R}$ is a given **one-time reward function**.

Note that

$$P(s_{\cdot}, a_{\cdot} | \theta_{\cdot}) = \prod_{t=0}^{T-1} \{P(s_t|s_{t-1}, a_{t-1})P(a_t|s_t, \theta_t)\} . \quad (61.6)$$

Define the **all times reward** Σ by

$$\Sigma(s., a.) = \sum_{t=0}^{T-1} \gamma^t r(s_t, a_t) . \quad (61.7)$$

Here $0 < \gamma < 1$. γ , called the **discount rate**, is included to assure convergence of Σ when $T \rightarrow \infty$. If $r(s_t, a_t) < K$ for all t , then $\Sigma < K \frac{1}{1-\gamma}$.

Define the **objective (i.e. goal) function** $E\Sigma(\theta.)$ by

$$E\Sigma(\theta.) = E_{\underline{s}.,\underline{a}.|\theta.} \Sigma(\underline{s}.,\underline{a}.) = \sum_{s.,a.} P(s.,a.|\theta.) \Sigma(s.,a.) \quad (61.8)$$

The goal of RL is to maximize the objective function over its parameters $\theta..$. The parameters θ^* . that maximize the objective function are the optimum strategy:

$$\theta.^* = \operatorname{argmax}_{\theta.} E\Sigma(\theta.) \quad (61.9)$$

Define a **future reward** for times $\geq t$ as:

$$\Sigma_{\geq t}((s_{t'}, a_{t'})_{t' \geq t}) = \sum_{t'=t}^{T-1} \gamma^{t'-t} r(s_{t'}, a_{t'}) \quad (61.10)$$

Define the following **expected conditional future rewards** (rewards for times $\geq t$, conditioned on certain quantities having given values):

$$v_t = v(s_t, a_t; \theta.) = E_{\underline{s}.,\underline{a}.|s_t,a_t,\theta.} [\Sigma_{\geq t}] \quad (61.11)$$

$$V_t = V(s_t; \theta.) = E_{\underline{s}.,\underline{a}.|s_t,\theta.} [\Sigma_{\geq t}] = E_{\underline{a}_t|s_t,\theta.} [v(s_t, \underline{a}_t; \theta.)] \quad (61.12)$$

v is usually called Q in the literature. We will refer to Q as v in order to follow a convention wherein an \underline{a}_t -average changes a lower case letter to an upper case one.

We will sometimes write $v(s_t, a_t)$ instead of $v(s_t, a_t; \theta.).$

Since $E\Sigma_{\geq t}$ only depends on $\theta_{\geq t}$, $v(s_t, a_t; \theta.) = v(s_t, a_t; \theta_{\geq t})$, and $V(s_t; \theta.) = V(s_t; \theta_{\geq t})$.

Note that the objective function $E\Sigma$ can be expressed in terms of v_0 by averaging over its unaveraged parameters:

$$E\Sigma(\theta.) = E_{\underline{s}_0,\underline{a}_0|\theta_0} v(\underline{s}_0, \underline{a}_0; \theta.) \quad (61.13)$$

Define a **one-time reward** and an **expected conditional one-time reward** as:

$$r_t = r(s_t, a_t) \quad (61.14)$$

$$R_t = R(s_t; \theta_t) = E_{\underline{a}_t|s_t,\theta_t} [r(s_t, \underline{a}_t)] . \quad (61.15)$$

Note that

$$\Sigma_{\geq t} = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots + \gamma^{T-1-t} r_{t+(T-1-t)} \quad (61.16)$$

$$= r_t + \gamma \Sigma_{\geq t+1}; \quad (61.17)$$

If we take $E_{s.,a.|s_t,a_t,\theta.}[\cdot]$ of both sides of Eq.(61.17), we get

$$v_t = r_t + \gamma E_{s_{t+1},a_{t+1}|\theta.}[v_{t+1}] . \quad (61.18)$$

If we take $E_{s.,a.|s_t,\theta.}[\cdot]$ of both sides of Eq.(61.17), we get

$$V_t = R_t + \gamma E_{s_{t+1}|\theta.}[V_{t+1}] . \quad (61.19)$$

Note that

$$\Delta r_t = r_t - R_t \quad (61.20)$$

$$= r_t - (V_t - \gamma E_{s_{t+1}|\theta.}[V_{t+1}]) \quad (61.21)$$

$$= r_t + \gamma E_{s_{t+1}|\theta.}[V_{t+1}] - V_t . \quad (61.22)$$

Define

$$\Delta v_t = v_t - V_t . \quad (61.23)$$

Note that

$$\Delta v_t = \Delta r_t . \quad (61.24)$$

Next, we will discuss 3 RL bnets

- exact RL bnet (exact, assumes policy is known)
- Actor-Critic RL bnet (approximate, assumes policy is known)
- Q function learning RL bnet (approximate, assumes policy is NOT known)

61.1 Exact RL bnet

An exact RL bnet is given by Fig.61.3.

Fig.61.3 is the same as Fig.61.2 but with more nodes added in order to optimize the policy parameters. The TPMs, printed in blue, for the nodes not already discussed in connection to bnet Fig.61.2, are as follows.

$$P(\theta_t|\theta.) = \delta(\theta_t, (\theta.)_t) \quad (61.25)$$

$$\forall(s_t, a_t) : P(v_t(s_t, a_t)|r_t, v_{t+1}(\cdot), \theta.) = \delta(v_t(s_t, a_t), r_t + \gamma E_{s_{t+1},a_{t+1}|\theta.}[v_{t+1}]) \quad (61.26)$$

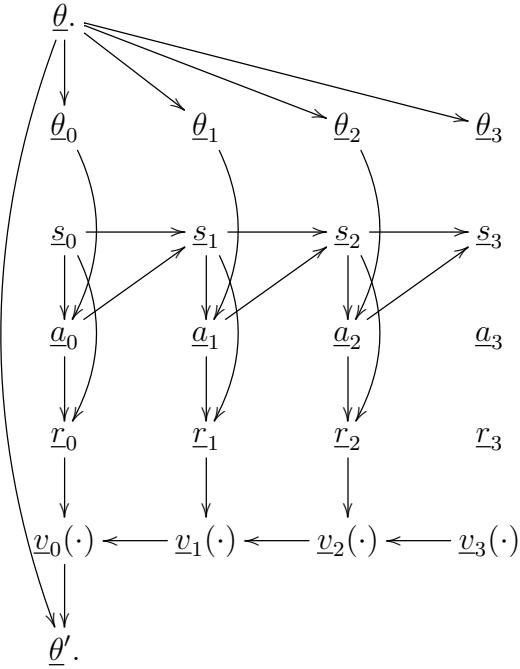


Figure 61.3: Exact RL bnet. $v_t(\cdot)$ means the array $[v_t(s_t, a_t)]_{\forall s_t, a_t}$. The following arrows are implicit: for all t , arrow from $\underline{\theta}_.$ $\rightarrow \underline{v}_t(\cdot)$. We did not draw those arrows so as not to clutter the diagram.

$$P(\theta' | \theta, v_0(\cdot)) = \delta(\theta', \theta + \alpha \partial_{\theta} \underbrace{E_{\underline{s}_0, \underline{a}_0 | \theta_0} v(\underline{s}_0, \underline{a}_0; \theta)}_{E\Sigma(\theta)}) \quad (61.27)$$

$\alpha > 0$ is called the **learning rate**. This method of improving θ . is called gradient ascent.

Concerning the gradient of the objective function, note that

$$\partial_{\theta_t} E\Sigma(\theta) = \sum_{s., a.} \partial_{\theta_t} P(s., a. | \theta.) \Sigma(s., a.) \quad (61.28)$$

$$= \sum_{s., a.} P(s., a. | \theta.) \partial_{\theta_t} \ln P(s., a. | \theta.) \Sigma(s., a.) \quad (61.29)$$

$$= E_{\underline{s}, \underline{a} | \theta.} \{ \partial_{\theta_t} \ln P(a_t | s_t, \theta_t) \Sigma(s., a.) \} . \quad (61.30)$$

If we run the agent $nsam(\vec{s}_t)$ times and obtain samples $s_t[i], a_t[i]$ for all t and for $i = 0, 1, \dots, nsam(\vec{s}_t) - 1$, we can express this gradient as follows:

$$\partial_{\theta_t} E\Sigma(\theta) \approx \frac{1}{nsam(\vec{s}_t)} \sum_i \sum_{t=0}^{T-1} \partial_{\theta_t} \ln P(a_t[i] | s_t[i], \theta_t) r(s_t[i], a_t[i]) . \quad (61.31)$$

The exact RL bnet Fig.61.3 is difficult to use to calculate the optimum parameters θ^* . The problem is that \underline{s}_t propagates towards the future and the $\underline{v}_t(\cdot)$ propagates towards the past, so we don't have a Markov Chain with a chain link for each t (i.e., a dynamical bnet) in the episode time direction. Hence, people have come up with approximate RL bnets that are doubly dynamical (i.e., dynamical along the episode time and episode number axes.) We discuss some of those approximate RL bnets next.

61.2 Actor-Critic RL bnet

For the actor-critic RL bnet, we approximate Eq.(61.31) by

$$\partial_{\theta_t} E\Sigma(\theta) \approx \frac{1}{nsam(\vec{s})} \sum_i \sum_{t=0}^{T-1} \underbrace{\partial_{\theta_t} \ln P(a_t[i] | s_t[i], \theta_t)}_{Actor} \underbrace{\Delta r_t(s_t[i], a_t[i])}_{Critic} \quad (61.32)$$

The actor-critic RL bnet is given by Fig.61.4. This bnet is approximate and assumes that the policy is known. The TPMs, printed in blue, for this bnet, are as follows.

$$P(\theta_t) = \text{given} \quad (61.33)$$

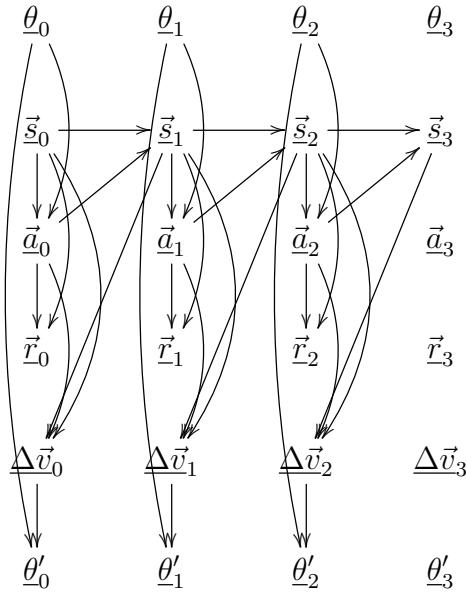


Figure 61.4: Actor-Critic RL bnet.

$$P(s_t[i] | s_{t-1}[i], a_{t-1}[i]) = \text{ given} \quad (61.34)$$

$$P(a_t[i] | s_t[i], \theta_t) = \text{ given} \quad (61.35)$$

$$P(r_t[i] | s_t[i], a_t[i]) = \delta(r_t[i], r(s_t[i], a_t[i])) \quad (61.36)$$

$r : S_{\underline{s}} \times S_{\underline{a}} \rightarrow \mathbb{R}$ is given.

$$P(\Delta v_t[i] | s_t[i], a_t[i], s_{t+1}[i]) = \delta(\Delta v_t[i], r(s_t[i], a_t[i]) + \gamma \hat{V}(s_{t+1}[i]; \phi') - \hat{V}(s_t[i]); \phi) . \quad (61.37)$$

$$P(\theta'.) = \delta(\theta'., \theta_t + \alpha \partial_{\theta_t} \sum_i \ln P(a_t[i] | s_t[i], \theta_t) \Delta v_t[i]) \quad (61.38)$$

$\hat{V}(s_t[i]; \phi)$ is obtained by curve fitting (see Chapter 5) using samples $(s_t[i], a_t[i])$ $\forall t, i$ with

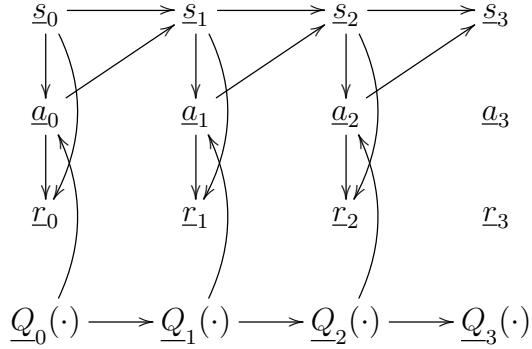


Figure 61.5: Q function learning RL bnet.

$$y[i] = \sum_{t'=t}^T r(s_{t'}[i], a_{t'}[i]) \quad (61.39)$$

and

$$\hat{y}[i] = \hat{V}(s_t[i]; \phi) . \quad (61.40)$$

Eq.(61.39) is an approximation because $(s_{t'}, a_{t'})_{t'>t}$ are averaged over in the exact expression for $V(s_t)$. $\hat{V}(s_{t+1}[i]); \phi'$ is obtained in the same way as $\hat{V}(s_t[i]); \phi$ but with t replaced by $t + 1$ and ϕ by ϕ' .

61.3 Q function learning RL bnet

The Q-function learning RL bnet is given by Fig.61.5. This bnet is approximate and assumes that the policy is NOT known. The TPMs, printed in blue, for this bnet, are as follows. (Remember that $Q = v$).

$$P(s_t | s_{t-1}, a_{t-1}) = \text{given} \quad (61.41)$$

$$P(a_t | s_t, v_t(\cdot)) = \delta(a_t, \operatorname{argmax}_a v_t(s_t, a)) \quad (61.42)$$

$$P(r_t | s_t, a_t) = \delta(r_t, r(s_t, a_t)) \quad (61.43)$$

$r : S_s \times S_a \rightarrow \mathbb{R}$ is given.

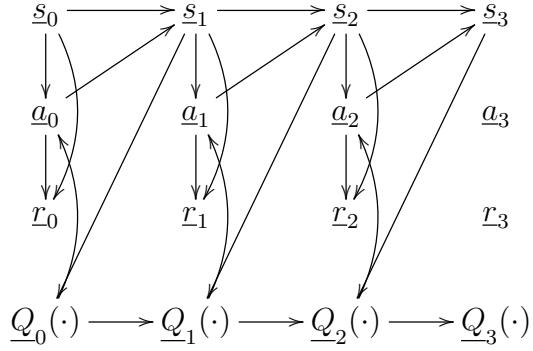


Figure 61.6: Q function learning RL bnet. Same as Fig.61.5 but with new arrow passing \$s_t\$ to \$Q_{t-1}\$.

$$\begin{aligned} \forall(s_t, a_t) : P(v_t(s_t, a_t) | v_{t-1}(\cdot)) = \\ = \delta(v_t(s_t, a_t), r(s_t, a_t) + \gamma \max_a E_{s_{t+1}|s_t, a_t} v_{t-1}(s_{t+1}, a)) \end{aligned} \quad (61.44)$$

This value for \$v_t(s_t, a_t)\$ approximates \$v_t = r_t + \gamma E_{s_{t+1}|s_t, a_t} v_{t+1}\$.

Some people use the bnet of Fig.61.6 instead of Fig.61.5 and replace Eq.(61.44) by

$$\begin{aligned} \forall(s_t, a_t) : P(v_t(s_t, a_t) | s_{t+1}, v_{t-1}(\cdot)) = \\ = \delta(v_t(s_t, a_t), r(s_t, a_t) + \gamma \max_a v_{t-1}(s_{t+1}, a)) . \end{aligned} \quad (61.45)$$

Chapter 62

Reliability Box Diagrams and Fault Tree Diagrams

This chapter is based on Refs.[46] and [63].

In this chapter, we assume that reader is familiar with Boolean Algebra. See Chapter Notational Conventions and Preliminaries for a quick review of what we recommend that you know about Boolean Algebra to fully appreciate this chapter.

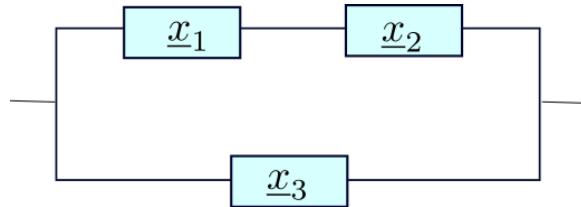


Figure 62.1: Example of rbox diagram.

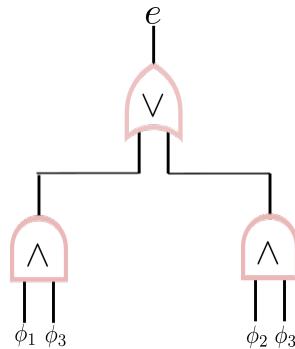


Figure 62.2: An ftree diagram equivalent to Fig.62.1. It represents $e = (\phi_1 \wedge \phi_3) \vee (\phi_2 \wedge \phi_3)$.

Complicated devices with a large number of components such as airplanes or rockets can fail in many ways. If their performance depends on some components

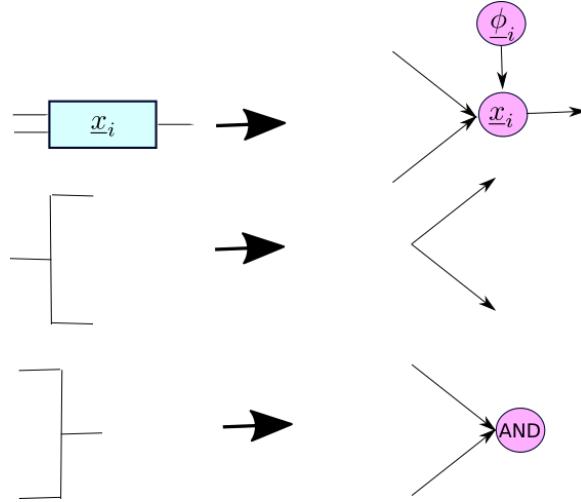


Figure 62.3: How to map an rbox diagram to a bnet.

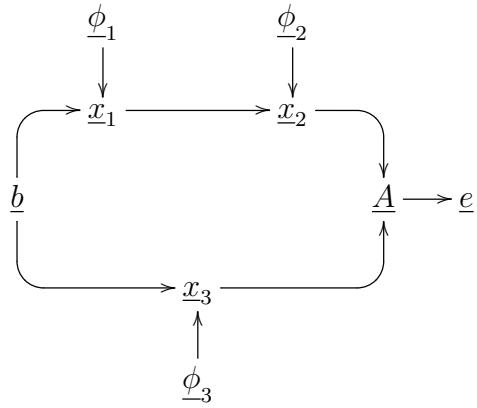


Figure 62.4: bnet corresponding the rbox diagram Fig.62.1.

working in series and one of the components in the series fails, this may lead to catastrophic failure. To avert such disasters, engineers use equivalent components connected in parallel instead of in series, thus providing multiple backup systems. They analyze the device to find its weak points and add backup capabilities there. They also estimate the average time to failure for the device.

The two most popular diagrams for finding the failure modes and their rates for large complicated devices are

- rbox diagrams = Reliability Box diagrams. See Fig.62.1 for an example.
- ftree diagrams = Fault Tree Diagrams. See Fig.62.2 for an example.

In an ftree diagram, several nodes might stand for the same component of a physical

device. In an rbox diagram, on the other hand, each node represents a distinct component in a device. Hence, rbox diagrams resemble the device they are addressing whereas ftree diagrams don't. Henceforth, we will refer to this desirable property as **physical resemblance**.

As we will show below with an example, it is pretty straightforward to translate an rbox to an ftree diagram. Going the other way, translating an ftree to an rbox diagram is much more difficult.

Next we will define a new kind of bnet that we will call a failure bnet that has physical resemblance. Then we will describe a simple method of translating (i.e., mapping) any rbox diagram to a failure bnet. Then we will show how a failure bnet can be used to do all the calculations that are normally done with an rbox or an ftree diagram. In that sense, failure bnets seem to afford all the benefits of both ftree and rbox diagrams.

A **failure bnet** contains nodes of 5 types, labeled \underline{b} , \underline{e} , \underline{x}_i , $\underline{\phi}_i$, and \underline{A}_i . All nodes have only two possible states $S = Success = 0$, $F = Failure = 1$.

1. The bnet has a beginning node labeled \underline{b} which is always set to success. The \underline{b} node and the $\underline{\phi}_i$ nodes are the only root nodes of the bnet.
2. The bnet has a single leaf node, the end node, labeled \underline{e} . \underline{e} is fixed. In rbox diagrams, $\underline{e} = S$ whereas in ftree diagrams, $\underline{e} = F$.
3. $\underline{x}^{nx} = (\underline{x}_0, \underline{x}_1, \dots, \underline{x}_{nx-1})$. $\underline{x}_i \in \{S, F\}$ for all i .

Suppose \underline{x}_i has parents $\underline{\phi}_i$ and $\underline{a}^{na} = (a_0, a_1, \dots, a_{na-1})$. Then the TPM of node \underline{x}_i is defined to be

$$P(x_i | \phi_i, a^{na}) = \delta(x_i, \phi_i \vee \vee_{i=0}^{na-1} a_i) \quad (62.1)$$

4. For each node \underline{x}_i , the bnet has a “performance” root node $\underline{\phi}_i \in \{0, 1\}$ with an arrow pointing from it to \underline{x}_i (i.e., $\underline{\phi}_i \rightarrow \underline{x}_i$). For all i ,

$$P(\phi_i) = \epsilon_i \delta(\phi_i, F) + \bar{\epsilon}_i \delta(\phi_i, S). \quad (62.2)$$

ϵ_i is the failure probability and $\bar{\epsilon}_i = 1 - \epsilon_i$ is the success probability. We name the failure probability ϵ_i because it is normally very small. It is usually set to $1 - e^{-\lambda_i t} \approx \lambda_i t$ when $\lambda_i t \ll 1$, where λ_i is the failure rate for node \underline{x}_i and t stands for time. The rblock literature usually calls $\bar{\epsilon}_i = R_i$ the **reliability** of node \underline{x}_i , and $\epsilon_i = (1 - R_i) = F_i$ its **unreliability**.

5. The nodes $\underline{A}_i \in \{0, 1\}$ are simply AND gates. If \underline{A}_i has inputs $\underline{y}^{ny} = (\underline{y}_0, \underline{y}_1, \dots, \underline{y}_{ny-1})$, then the TPM of \underline{A}_i is

$$P(A_i|y^{ny}) = \delta(A_i, \wedge_{i=0}^{ny-1} y_i) . \quad (62.3)$$

An instance (instantiation) of a bnet is the bnet with all nodes set to a specific state. A **realizable instance (r-instance)** of a bnet is one which has non-zero probability.

Fig.62.3 shows how to translate any rbox diagram to a failure bnet. To illustrate this procedure, we translated the rbox diagram Fig.62.1 into the failure bnet Fig.62.4.

For the failure bnet Fig.62.4, one has:

$$\begin{aligned} P(b) &= \mathbb{1}(b = 0) \\ P(x_1|\phi_1, b) &= \mathbb{1}(x_1 = \phi_1 \vee b) \\ P(x_2|\phi_2, x_1) &= \mathbb{1}(x_2 = \phi_2 \vee x_1) \\ P(x_3|\phi_3, b) &= \mathbb{1}(x_3 = \phi_3 \vee b) \\ P(A|x_2, x_3)e &= \mathbb{1}(x_2 \wedge x_3) \\ P(e|A) &= \mathbb{1}(e = A) \end{aligned} . \quad (62.4)$$

Therefore, all r-instances of this bnet must satisfy

$$e = (\phi_1 \vee \phi_2) \wedge \phi_3 \quad (62.5)$$

$$= (\phi_1 \wedge \phi_3) \vee (\phi_2 \wedge \phi_3) . \quad (62.6)$$

Eq.(62.6) proves that Fig.62.2 is indeed a representation of Fig.62.1.

Next, we consider r-instances of this bnet for two cases: $e = S$ and $e = F$.

- **rblock analysis:** $e = S = 0$.

Table 62.1 shows the probability of all possible r-instances that end in success for the failure bnet Fig.62.4. (These r-instances are the main focus of rblock analysis). The first 4 of those probabilities (those with $\phi_3 = 0$) sum to $\bar{\epsilon}_3$ so the sum $P(e = S)$ of all 5 is

$$P(e = S) = \bar{\epsilon}_3 + \bar{\epsilon}_1 \bar{\epsilon}_2 \epsilon_3 , \quad (62.7)$$

or, expressing it in reliability language in which $\bar{\epsilon} = R$,

$$P(e = S) = R_3 + R_1 R_2 \bar{R}_3 . \quad (62.8)$$

- **ftree analysis:** $e = F = 1$.

Table 62.2 shows the probability of all possible r-instances that end in failure for the failure bnet Fig.62.4. (These r-instances are the main focus of ftree analysis). If we set $\epsilon_i = \epsilon$ and $\bar{\epsilon}_i \approx 1$ for $i = 1, 2, 3$, then the first two of

instance	probability
<p>Diagram of a belief network instance with three nodes x_1, x_2, and x_3. Node x_1 has two outgoing edges: one to x_2 labeled 0 and one to x_3 labeled 0. Node x_2 has two outgoing edges: one to x_3 labeled 1 and one to a hidden node A labeled $\overline{A} \Rightarrow 0$. Node x_3 has one outgoing edge to A labeled 0.</p>	$\bar{\epsilon}_1 \epsilon_2 \bar{\epsilon}_3$
<p>Diagram of a belief network instance with three nodes x_1, x_2, and x_3. Node x_1 has two outgoing edges: one to x_2 labeled 1 and one to x_3 labeled 0. Node x_2 has two outgoing edges: one to x_3 labeled 0 and one to a hidden node A labeled $\overline{A} \Rightarrow 0$. Node x_3 has one outgoing edge to A labeled 0.</p>	$\epsilon_1 \bar{\epsilon}_2 \bar{\epsilon}_3$
<p>Diagram of a belief network instance with three nodes x_1, x_2, and x_3. Node x_1 has two outgoing edges: one to x_2 labeled 1 and one to x_3 labeled 0. Node x_2 has two outgoing edges: one to x_3 labeled 1 and one to a hidden node A labeled $\overline{A} \Rightarrow 0$. Node x_3 has one outgoing edge to A labeled 0.</p>	$\epsilon_1 \epsilon_2 \bar{\epsilon}_3$
<p>Diagram of a belief network instance with three nodes x_1, x_2, and x_3. Node x_1 has two outgoing edges: one to x_2 labeled 0 and one to x_3 labeled 0. Node x_2 has two outgoing edges: one to x_3 labeled 0 and one to a hidden node A labeled $\overline{A} \Rightarrow 0$. Node x_3 has one outgoing edge to A labeled 0.</p>	$\bar{\epsilon}_1 \bar{\epsilon}_2 \bar{\epsilon}_3$
<p>Diagram of a belief network instance with three nodes x_1, x_2, and x_3. Node x_1 has two outgoing edges: one to x_2 labeled 0 and one to x_3 labeled 1. Node x_2 has two outgoing edges: one to x_3 labeled 0 and one to a hidden node A labeled $\overline{A} \Rightarrow 0$. Node x_3 has one outgoing edge to A labeled 0.</p>	$\bar{\epsilon}_1 \bar{\epsilon}_2 \epsilon_3$

Table 62.1: Probabilities of all possible r-instances with $e = S = 0$ for failure bnet Fig.62.4.

instance	probability
	$\bar{\epsilon}_1 \epsilon_2 \epsilon_3$
	$\epsilon_1 \bar{\epsilon}_2 \epsilon_3$
	$\epsilon_1 \epsilon_2 \epsilon_3$

Table 62.2: Probabilities of all possible r-instances with $e = F = 1$ for the failure bnet Fig.62.4.

those r-instances have probabilities of $order(\epsilon^2)$ and the third has probability of $order(\epsilon^3)$. The two lowest order ($order(\epsilon^2)$) r-instances are called the “minimal cut sets” of the ftree. We will have more to say about minimal cut sets later on. For now, just note from Eq.(62.6) that the ftree Fig.62.2 is just the result of joining together with ORs two expressions, one for each of the two minimal cut sets.

More general \underline{x}_i .

Failure bnets can actually accommodate \underline{x}_i nodes of a more general kind than what we first stipulated. Here are some possibilities:

For any $a^n \in \{0, 1\}^n$, let

$$len(a^n) = \sum_i a_i \quad (62.9)$$

- **OR gate**

$$P(x_i|\phi_i, a^{na}) = \delta(x_i, \phi_i \vee \vee_j a_j) \quad (62.10)$$

$$= \delta(x_i, \phi_i \vee \mathbb{1}(\text{len}(a^{na}) > 0)) \quad (62.11)$$

- **AND gate**

$$P(x_i|\phi_i, a^{na}) = \delta(x_i, \phi_i \vee \wedge_j a_j) \quad (62.12)$$

$$= \delta(x_i, \phi_i \vee \mathbb{1}(\text{len}(a^{na}) = na)) \quad (62.13)$$

- **Fail if least K failures (less than K successes)**

$$P(x_i|\phi_i, a^{na}) = \delta(x_i, \phi_i \vee \mathbb{1}(\text{len}(a^{na}) \geq K)) \quad (62.14)$$

- **Fail if less than K failures (at least K successes)**

$$P(x_i|\phi_i, a^{na}) = \delta(x_i, \phi_i \vee \mathbb{1}(\text{len}(a^{na}) < K)) \quad (62.15)$$

- **Fail if exactly one failure**

$$P(x_i|\phi_i, a^{na}) = \delta(x_i, \phi_i \vee \mathbb{1}(\text{len}(a^{na}) = 1)) \quad (62.16)$$

This equals an XOR (exclusive OR) gate when $na = 2$.

- **General gate**

$$f : \{0, 1\}^{na} \rightarrow \{0, 1\}$$

$$P(x_i|\phi_i, a^{na}) = \delta(x_i, \phi_i \vee f(a^{na})) \quad (62.17)$$

62.1 Minimal Cut Sets

Suppose $x \in \{0, 1\}$ and $f : \{0, 1\} \rightarrow \{0, 1\}$. Then by direct evaluation, we see that

$$f(x) = [\bar{x}f(0)] \vee [xf(1)]. \quad (62.18)$$

Let

$$\begin{aligned} !x &= 1 - x, \\ !^0 x &= x, \\ !^1 x &= !x \end{aligned} \tag{62.19}$$

Then Eq.(62.18) can be rewritten as

$$f(x) = \vee_{a \in \{0,1\}} [(!^{\bar{a}} x) f(a)] . \tag{62.20}$$

Now suppose $x^n \in \{0,1\}^n$ and $f : \{0,1\}^n \rightarrow \{0,1\}$. Eq.(62.20) generalizes to

$$f(x^n) = \vee_{a^n \in \{0,1\}^n} \left[\prod_i (!^{\bar{a}_i} x_i) f(a^n) \right] . \tag{62.21}$$

Eq.(62.21) is called an **ors-of-ands** normal form expansion. There is also an **ands-of-orss** normal form expansion obtained by swapping multiplication and \vee in Eq.(62.21), but we won't need it here.

A **cut set** is a set of ϕ_i 's such that if they are all equal to F , then $e = F$ for all the r-instances. A **minimal cut set** is a cut set such that there are no larger cut sets that contain it. From the failure bnet, we can always find a function $f : \{0,1\}^{nx} \rightarrow \{0,1\}$ such that $e = f(\phi^{nx})$ for all the r-instances. We did that for our example failure bnet and obtained Eq.(62.6). We can then express $f(\phi^{nx})$ as an ors-of-ands expansion to find all the minimal cut sets. The ands terms in that ors-of-ands expansion each gives a different minimal cut set, after some simplification. The ors-of-ands expression is not unique and it may be necessary to simplify (using the Boolean Algebra identities given in Chapter Notational Conventions and Preliminaries) to remove those redundancies.

Chapter 63

Restricted Boltzmann Machines

In what follows, we will abbreviate "restricted Boltzmann machine" by rebo.

Let

$$v \in \{0, 1\}^{nv}$$

$$h \in \{0, 1\}^{nh}$$

$b \in \mathbb{R}^{nv}$ (mnemonic, v and b sound the same)

$$a \in \mathbb{R}^{nh}$$

$$W^{v|h} \in \mathbb{R}^{nv \times nh}$$

Energy:

$$E(v, h) = -(b^T v + a^T h + v^T W^{v|h} h) \quad (63.1)$$

Boltzmann distribution:

$$P(v, h) = \frac{e^{-E(v, h)}}{Z} \quad (63.2)$$

Partition function:

$$Z = \sum_{v, h} e^{-E(v, h)} = Z(a, b, W^{v|h}) \quad (63.3)$$

$$P(v|h) = \frac{e^{b^T v + a^T h + v^T W^{v|h} h}}{\sum_v e^{b^T v + a^T h + v^T W^{v|h} h}} \quad (63.4)$$

$$= \frac{e^{b^T v + v^T W^{v|h} h}}{\sum_v e^{b^T v + v^T W^{v|h} h}} \quad (63.5)$$

$$= \prod_i \frac{e^{v_i(b_i + \sum_j W_{i,j}^{v|h} h_j)}}{\sum_{v_i=0,1} e^{v_i(b_i + \sum_j W_{i,j}^{v|h} h_j)}} \quad (63.6)$$

$$= \prod_i P(v_i|h) \quad (63.7)$$

$$P(v_i|h) = \frac{e^{v_i(b_i + \sum_j W_{i,j}^{v|h} h_j)}}{Z_i(h)} \quad (63.8)$$

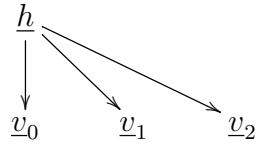


Figure 63.1: bnet for a Restricted Boltzmann Machine (rebo) with $nv = 3$

Eq.(63.8) implies that a rebo can be represented by the bnet Fig.63.1.

Let

$$x_i = b_i + \sum_j W_{ij}^{vh} h_j . \quad (63.9)$$

Then

$$P(v_i = 1|h) = \frac{e^{x_i}}{1 + e^{x_i}} \quad (63.10)$$

$$= \frac{1}{1 + e^{-x_i}} \quad (63.11)$$

$$= \text{smoid}(x_i) . \quad (63.12)$$

One could also expand the node h in Fig.63.1 into nh nodes. But note that $P(h) \neq \prod_j P(h_j)$ so there would be arrows among the h_j nodes.

Note that the rebo bnet is a special case of Naive Bayes (See Chapter 49) with $v_i, h_i \in \{0, 1\}$ and specific $P(h)$ and $P(v_i|h)$ node matrices.

Chapter 64

ROC curves

This chapter is based on Ref.[121].

ROC stands for **Receiver Operating Characteristic**. ROC curves are used in binary classification (BC).

To do BC, we are given the value $x \in \mathbb{R}$ for an individual. From this, we want to decide whether that individual has $a = 0$ or $a = 1$. The decision will depend on the value of a **threshold parameter** $\tau \in \mathbb{R}$.

$$\underline{x} \longleftarrow a$$

Figure 64.1: bnet for BC.

Fig.64.1 shows the bnet used for BC.

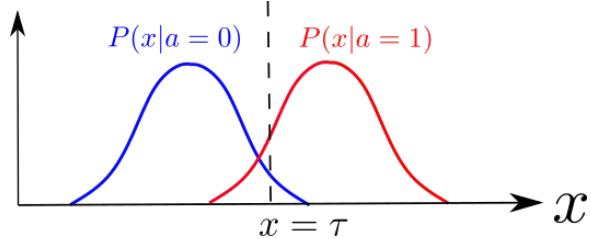


Figure 64.2: x -distribution for two hypotheses $a = 0, 1$.

Fig.64.2 is a plot of $P(x|a)$, i.e., the TPM for node \underline{x} of the bnet in Fig.64.1. Whereas a is binary, x is continuous. But we can replace x by a binary variable

$$b = \mathbb{1}(x > \tau). \quad (64.1)$$

$P(b|a)$ for $b, a \in \{0, 1\}$ is called the **confusion matrix** or **contingency table** for BC. The confusion matrix can be calculated from the TPM $P(x|a)$. Fig.64.3 illustrates

		Actual Value (a)	
		0	1
Predicted Value (x)	0	$TNR \triangleleft$	$FNR \curvearrowright$
	1	$FPR \curvearrowleft$	$TPR \triangleup$

Figure 64.3: The confusion matrix $P(b|a)$ for BC.

the confusion matrix $P(b|a)$ for BC. In that figure, the rates R are defined as follows.¹

- **True Negative Rate (TNR)**

$$R_{0|0}(\tau) = P(x < \tau | a = 0) = \int_{x < \tau} dx P(x | a = 0) \quad (64.2)$$

- **False Positive Rate (FPR)**

$$R_{1|0}(\tau) = 1 - R_{0|0}(\tau) \quad (64.3)$$

$$= P(x > \tau | a = 0) = \int_{x > \tau} dx P(x | a = 0) \quad (64.4)$$

In Hypothesis Testing, $R_{1|0}$ is called the **p-value that $x > \tau$ assuming curve 0 is the null hypothesis**.

- **False Negative Rate (FNR)**

$$R_{0|1}(\tau) = P(x < \tau | a = 1) = \int_{x < \tau} dx P(x | a = 1) \quad (64.5)$$

In Hypothesis Testing, $R_{0|1}$ is called the **p-value that $x < \tau$ assuming curve 1 is the null hypothesis**.

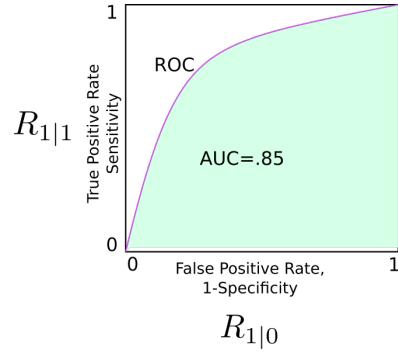
- **True Positive Rate (TPR)**

$$R_{1|1}(\tau) = 1 - R_{0|1}(\tau) \quad (64.6)$$

$$= P(x > \tau | a = 1) = \int_{x > \tau} dx P(x | a = 1) \quad (64.7)$$

The **Receiver Operating Characteristic (ROC)** is a parametric plot with $X = R_{1|0}(\tau)$ and $Y = R_{1|1}(\tau)$, where $\tau \in \mathbb{R}$. The **Area Under the Curve (AUC)** is the area under the ROC. Fig.64.4 shows an example of a ROC and its AUC.

¹I find the notation $x|a$ where $x, a \in \{0, 1\}$ much clearer than $\alpha\beta$ where $\alpha = T, F$ and $\beta = N, P$. Note that $\alpha = \mathbb{1}(x = a)$ and $\beta = x$, if we identify $0 = F = N$ and $1 = T = P$.



$$R_{1|0}$$

Figure 64.4: Example of ROC. Green shaded area is the AUC of the ROC.

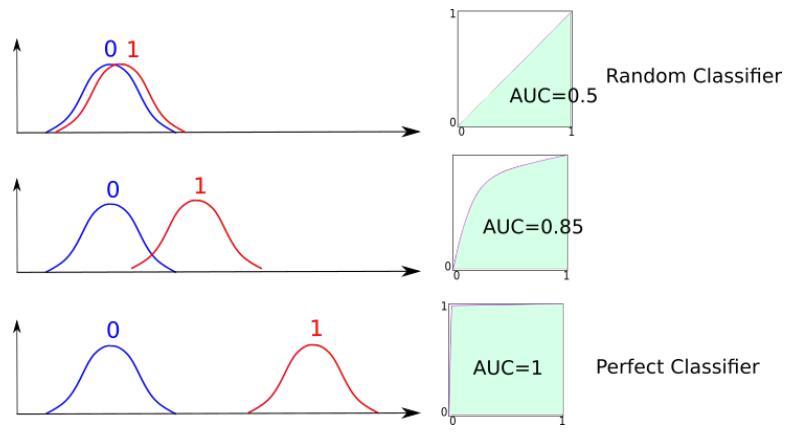


Figure 64.5: ROC curves for 3 different separations between the 0 and 1 x-distributions.

Fig.64.5 shows situations that give $AUC=.5$ (random classifier), $AUC=.85$, and $AUC=1$ (perfect classifier). It's also possible to get an $AUC \in [0, 0.5]$, but we will ignore those models because they are useless for BC.

Note that

$$AUC = \int_{x=0}^1 d\tau R_{1|1}(\tau) \frac{dR_{1|0}(\tau)}{d\tau} \quad (64.8)$$

$$= \int_{-\infty}^{-\infty} d\tau \left\{ \int_{-\infty}^{\infty} dx \mathbb{1}(x > \tau) P(x|a=1) \right\} (-1) P(x = \tau|a=0) \quad (64.9)$$

$$= \int_{-\infty}^{\infty} dx' \int_{-\infty}^{\infty} dx \mathbb{1}(x > x') P(x|a=1) P(x'|a=0) . \quad (64.10)$$

64.1 Terminology Table Adapted from Wikipedia Ref.[121]

Let $N_{x|a}$ be numbers (counts) so that

$$P(x|a) = \frac{N_{x|a}}{\sum_{x'} N_{x'|a}} \quad (64.11)$$

for all $x, a \in \{0, 1\}$.

condition positive (P): $N_{|1} = \sum_x N_{x|1}$, the number of real positive cases in the data

condition negative (N): $N_{|0} = \sum_x N_{x|0}$, the number of real negative cases in the data

true positive (TP): $N_{1|1}$, hit

true negative (TN): $N_{0|0}$, correct rejection

false positive (FP): $N_{1|0}$, false alarm, type I error or underestimation

false negative (FN): $N_{0|1}$, miss, type II error or overestimation

sensitivity, recall, hit rate, or true positive rate (TPR):

$$TPR = R_{1|1} = \frac{N_{1|1}}{N_{|1}} = \frac{N_{1|1}}{N_{1|1} + N_{0|1}} = 1 - R_{0|1} \quad (64.12)$$

specificity, selectivity or true negative rate (TNR):

$$TNR = R_{0|0} = \frac{N_{0|0}}{N_{|0}} = \frac{N_{0|0}}{N_{0|0} + N_{1|0}} = 1 - R_{1|0} \quad (64.13)$$

precision or positive predictive value (PPV):

$$PPV = \frac{N_{1|1}}{N_{1|1} + N_{1|0}} = 1 - FDR \quad (64.14)$$

negative predictive value (NPV):

$$NPV = \frac{N_{0|0}}{N_{0|0} + N_{0|1}} = 1 - FOR \quad (64.15)$$

miss rate or false negative rate (FNR):

$$FNR = R_{0|1} = \frac{N_{0|1}}{N_{|1}} = \frac{N_{0|1}}{N_{0|1} + N_{1|1}} = 1 - R_{1|1} \quad (64.16)$$

fall-out or false positive rate (FPR):

$$FPR = R_{1|0} = \frac{N_{1|0}}{N_{|0}} = \frac{N_{1|0}}{N_{1|0} + N_{0|0}} = 1 - R_{0|0} \quad (64.17)$$

false discovery rate (FDR):

$$FDR = \frac{N_{1|0}}{N_{1|0} + N_{1|1}} = 1 - PPV \quad (64.18)$$

false omission rate (FOR):

$$FOR = \frac{N_{0|1}}{N_{0|1} + N_{0|0}} = 1 - NPV \quad (64.19)$$

accuracy (ACC):

$$ACC = \frac{N_{1|1} + N_{0|0}}{N_{1|1} + N_{0|0}} = \frac{N_{1|1} + N_{0|0}}{N_{1|1} + N_{0|0} + N_{1|0} + N_{0|1}} \quad (64.20)$$

balanced accuracy (BA):

$$BA = \frac{R_{1|1} + R_{0|0}}{2} \quad (64.21)$$

F1 score is the harmonic mean of precision and sensitivity:

$$F_1 = 2 \times \frac{PPV \times R_{1|1}}{PPV + R_{1|1}} = \frac{2N_{1|1}}{2N_{1|1} + N_{1|0} + N_{0|1}} \quad (64.22)$$

Chapter 65

Scoring the Nodes of a Learned Bnet

Chapter 70 discusses how to learn a bnet from data. Many algorithms for doing this require scoring how well a particular bnet fits the data. This chapter is an introduction to such scoring.

Normally, each node of a bnet is scored separately, and then those node scores are summed to get the bnet score.

In this chapter, scores are defined so that a higher score means a better fit. By taking the negative of such a score, one can always get a score such that a lower score means a better fit.

There are 2 main types of bnet scores: Maximum Likelihood (ML) scores, and Shannon Information Theory (SIT) scores. ML scores consist of the log of a maximum likelihood function $P(\vec{x}|\theta)$ for i.i.d. samples $\vec{x} = (x^\sigma)_{\sigma=0,1,\dots,nsam-1}$, where $x^\sigma \sim P_{\underline{x}|\theta}(x|\theta)$:

$$\text{ML-score} = \ln(P(\vec{x}|\theta)) \quad (65.1)$$

$$= \ln \prod_{\sigma} P(x^\sigma | \theta) \quad (65.2)$$

$$= \sum_{\sigma} \ln P(x^\sigma | \theta) \quad (65.3)$$

$$\approx nsam \sum_x P(x|\theta) \ln P(x|\theta) \quad (65.4)$$

$$= -nsam H(P_{\underline{x}|\theta}) , \quad (65.5)$$

and SIT scores consist of a negative entropy:

$$\text{Info-score} = -H(P_{\underline{x}|\theta}) . \quad (65.6)$$

Thus, up to a factor of $nsam$, they are the same thing. Maximizing a log likelihood function for i.i.d. samples or minimizing the corresponding entropy, are the same

thing, and they both yield a good estimate of the hidden parameters θ .

65.1 Probability Distributions and Special Functions

While writing this chapter, I briefly consulted the following Wikipedia articles about the definitions and properties of certain probability distributions and special functions.

- Categorical Distribution, Ref.[77]
- Multinomial Distribution, Ref.[113]
- Dirichlet Distribution, Ref.[84]
- Multivariate Normal Distribution, Ref.[115]
- Beta function, Ref.[73]
- Multinomial Coefficients, Ref.[114]
- Gamma Function Ref.[89]

Here are a few results from those Wikipedia articles that we will use later on in this chapter.

Below, we will abbreviate $q_+ = \sum_i q_i$, and $q_\cdot = (q_0, q_1, \dots, q_{nq-1})$ for various quantities q

Gamma function. If $n > 0$ is an integer,

$$\Gamma(n+1) = n! \quad (65.7)$$

The **multivariate Beta function** is defined by

$$B(\alpha_\cdot) = \frac{\prod_k \Gamma(\alpha_k)}{\Gamma(\alpha_+)} \quad (65.8)$$

where $\alpha_k > 0$ for all k .

The **multinomial coefficient** is defined by

$$C(N_\cdot) = \frac{N_+!}{\prod_k N_k!} \quad (65.9)$$

where N_k are non-negative integers.

The **inverse of the multinomial coefficient** will be denoted by

$$CI(N_\cdot) = \frac{1}{C(N_\cdot)} = \frac{\prod_k N_k!}{N_+!} \quad (65.10)$$

The **Categorical Distribution** is defined by

$$Cat(x; \pi.) = \pi_x = \prod_k \pi_k^{\mathbb{1}(k=x)} \quad (65.11)$$

for $k, x \in S_x$, where $\pi.$ is a probability dist.(i.e., $\pi_k \geq 0$ for all k , and $\pi_+ = 1$).

The **Multinomial Distribution** is defined by

$$Mul(N.; \pi., N) = C(N.) , \quad (65.12)$$

where N_k is a non-negative integer for all k , $N_+ = N$, and $\pi.$ is a probability dist. $Mul()$ satisfies:

$$E[N_k] = N\pi_k . \quad (65.13)$$

The **Dirichlet Distribution** is defined by

$$Dir(\pi.; \alpha.) = \frac{1}{B(\alpha.)} \prod_k \pi_k^{\alpha_k - 1} \quad (65.14)$$

where $\alpha_k > 0$ for all k , and $\pi.$ is a probability dist. The $\alpha.$ are called **concentration parameters** or **hyperparameters**. $Dir()$ satisfies:

$$E[\pi_k] = \frac{N_k}{N_+} . \quad (65.15)$$

$Dir()$ is **conjugate prior** of $Mul()$

Note that

$$Mul(N.; \pi., N) Dir(\pi.; \alpha.) = \mathcal{K}(N., \alpha.) Dir(\pi.; N. + \alpha.) , \quad (65.16)$$

where

$$\mathcal{K}(N., \alpha.) = \frac{B(N. + \alpha.)}{CI(N.)B(\alpha.)} . \quad (65.17)$$

$Dir()$ is replaceable by a $Mul()$ for large concentration parameters

Note that if N_k is a positive integer and $\alpha_k = N_k + 1$ for all k , then

$$Dir(\pi.; \alpha_k = N_k + 1) = C(N.) \prod_k \pi_k^{N_k} \quad (65.18)$$

$$= Mul(N.; \pi., N_+) . \quad (65.19)$$

65.2 Single node with no parents

In this section, we consider a learned bnet consisting of a single node with no parents. We will consider arbitrary learned bnets in the next section. But we start with this simplified case so as to reduce the number of indices in most quantities from 3 to 1. All the results that we derive in this section will be used in the next section after adding the extra indices. This way, we will avoid carrying the extra indices throughout the intermediate steps of many derivations.

For state $k \in \{0, 1, \dots, nk - 1\}$ of a single node \underline{x} , let

\underline{N}_k = current count number (an integer, data)

$\underline{\pi}_.$ = a probability dist, the TPM for the node

$\underline{\alpha}_k$ = prior count number

$$\underline{N}_. \longleftrightarrow \underline{\pi}_. \longleftrightarrow \underline{\alpha}_.$$

Figure 65.1: For a bnet consisting of a single node with no parents, this is a Markov chain of current counts ($\underline{N}_.$), TPM ($\underline{\pi}_.$), and prior counts ($\underline{\alpha}_.$).

Consider the Markov chain bnet of Fig.65.1 with the following TPMs, printed in blue.

$$P(N_+ | \underline{\pi}_.) = \text{Mul}(N_+; \underline{\pi}_., N_+) \quad (65.20)$$

$$P(\underline{\pi}_+ | \underline{\alpha}_.) = \text{Dir}(\underline{\pi}_+; \underline{\alpha}_.) \quad (65.21)$$

It follows that

$$P(N_+, \underline{\pi}_+ | \underline{\alpha}_.) = P(N_+ | \underline{\pi}_.) P(\underline{\pi}_+ | \underline{\alpha}_.) \quad (65.22)$$

$$= \text{Mul}(N_+; \underline{\pi}_., N_+) \text{Dir}(\underline{\pi}_+; \underline{\alpha}_.) \quad (65.23)$$

$$= \mathcal{K}(N_+, \underline{\alpha}_.) \text{Dir}(\underline{\pi}_+; N_+ + \underline{\alpha}_.) \quad (65.24)$$

From Eq.(65.15) for the expected value of $\text{Dir}()$, we get

$$\hat{\pi}_+ = E[\underline{\pi}_+] = \frac{N_+ + \underline{\alpha}_+}{N_+ + \underline{\alpha}_+}. \quad (65.25)$$

Integrating both sides of Eq.(65.24) over $\underline{\pi}_.$, we find that

$$P(N_+ | \underline{\alpha}_.) = \mathcal{K}(N_+, \underline{\alpha}_.). \quad (65.26)$$

If $N_k \gg 1$ for all k , then the $\text{Dir}()$ in Eq.(65.24) can be replaced by a $\text{Mul}()$

$$P(N., \pi.| \alpha.) \approx \mathcal{K}(N., \alpha.) Mul(N. + \alpha.; \pi., N_+ + \alpha_+) . \quad (65.27)$$

Therefore,

$$P(N.| \pi., \alpha.) = \frac{P(N., \pi.| \alpha.)}{P(N.| \alpha.)} \quad (65.28)$$

$$= Mul(N. + \alpha.; \pi., N_+ + \alpha_+) . \quad (65.29)$$

Claim 91

$$\ln P(N.| \hat{\pi}., \alpha.) = -(N_+ + \alpha_+) H \left(\frac{N. + \alpha.}{N_+ + \alpha_+} \right) + \ln C(N. + \alpha.) \quad (65.30)$$

$$> -(N_+ + \alpha_+) H \left(\frac{N. + \alpha.}{N_+ + \alpha_+} \right) - \frac{1}{2}(nk - 1) \ln N_+ \quad (65.31)$$

proof:

$$\ln P(N.| \hat{\pi}., \alpha.) = \sum_k (N_k + \alpha_k) \ln \hat{\pi}_k + \ln C(N. + \alpha.) \quad (65.32)$$

$$= \sum_k (N_k + \alpha_k) \ln \frac{N_k + \alpha_k}{N_+ + \alpha_+} + \ln C(N. + \alpha.) \quad (65.33)$$

$$= -(N_+ + \alpha_+) H \left(\frac{N. + \alpha.}{N_+ + \alpha_+} \right) + \ln C(N. + \alpha.) \quad (65.34)$$

Recall Stirling's approximation of a factorial, valid for large integers n :

$$\ln n! \approx (n + \frac{1}{2}) \ln n - n . \quad (65.35)$$

Assume $N_k \gg 1$ for all k . Applying Stirling's approximation to all factorials in $C(N)$, we get

$$\ln C(N.) \approx (N_+ + \frac{1}{2}) \ln N_+ - N_+ - \sum_k \left[(N_k + \frac{1}{2}) \ln N_k - N_k \right] \quad (65.36)$$

$$= (N_+ + \frac{1}{2}) \ln N_+ - \sum_k (N_k + \frac{1}{2}) \ln N_k . \quad (65.37)$$

Next assume that

$$N_k \approx \frac{N_+}{nk} . \quad (65.38)$$

Then

$$\ln C(N.) = (N_+ + \frac{1}{2}) \ln N_+ - nk(\frac{N_+}{nk} + \frac{1}{2})[\ln N_+ - \ln nk] \quad (65.39)$$

$$= -\frac{1}{2}(nk - 1) \ln N_+ + (N_+ + \frac{nk}{2}) \ln nk \quad (65.40)$$

$$> -\frac{1}{2}(nk - 1) \ln N_+ . \quad (65.41)$$

QED

65.3 Multiple nodes with any number of parents

In the previous section, we considered a bnet consisting of a single node with no parents, so we only needed a single index k for the states of the single node. In this section, we consider an arbitrary bnet with multiple nodes each of which may have multiple parents. Most of the results in the previous section are valid for the general case if we make the following replacements: $\pi.$ $\rightarrow \underline{\pi^i}_{|\mu}$ $N.$ $\rightarrow \underline{N^i}_{,\mu}$ $\alpha.$ $\rightarrow \underline{\alpha^i}_{,\mu}$. Upon this replacement, Fig.65.1 becomes Fig.65.2. The TPMs, printed in blue, of the new Markov chain, are as follows:

$$\underline{N^i}_{,\mu} \longleftarrow \underline{\pi^i}_{|\mu} \longleftarrow \underline{\alpha^i}_{,\mu}$$

Figure 65.2: Generalization of Fig.65.1. For a bnet with multiple nodes each of which may have multiple parents, this is a Markov chain of current counts ($\underline{N^i}_{,\mu}$), TPM ($\underline{\pi^i}_{|\mu}$), and prior counts ($\underline{\alpha^i}_{,\mu}$) .

$$P(\underline{N^i}_{,\mu} | \underline{\pi^i}_{|\mu}) = \text{Mul}(\underline{N^i}_{,\mu}; \underline{\pi^i}_{|\mu}, \underline{N^i}_{+, \mu}) \quad (65.42)$$

$$P(\underline{\pi^i}_{|\mu} | \underline{\alpha^i}_{,\mu}) = \text{Dir}(\underline{\pi^i}_{|\mu}; \underline{\alpha^i}_{,\mu}) \quad (65.43)$$

In these TPMs,

$i \in S_i = \{0, 1, \dots, ni - 1\}$ = node index

$\underline{x}.$ = $(\underline{x}_i)_{i \in S_i}$ = the nodes of the learned bnet.

$k \in S_{k^i} = \{0, 1, \dots, nk^i - 1\}$ = states of node x_i

$\mu \in S_{\mu^i} = \{0, 1, \dots, n\mu^i - 1\}$ = states of parents of node \underline{x}_i .

In the previous section, we assumed a single node ($ni = 1$) with no parents ($n\mu^0 = 1$) so that we could drop the i, μ indices. In this section, we eliminate that restriction.

It is convenient to define the magnitude of a bnet B to equal the sum over nodes of the number of free parameters in each TPM:

$$|B| = \sum_i (nk^i - 1)n\mu^i . \quad (65.44)$$

Suppose that we are given $nsam$ samples $\vec{x}_i = (\underline{x}_i^\sigma)_{\sigma=0,1,\dots,nsam-1}$ of our learned bnet. The count numbers $N_{k,\mu}^i$ are defined in terms of those samples as follows:

$$N_{k,\mu}^i = \sum_\sigma \mathbb{1}(\underline{x}_i^\sigma = k, pa(\underline{x}_i^\sigma) = \mu) . \quad (65.45)$$

It is also convenient to defined count number ratios

$$N_{k|\mu}^i = \frac{N_{k,\mu}^i}{N_{+,\mu}^i} . \quad (65.46)$$

Note that $N_{k,\mu}^i$ is a positive integer whereas $N_{k|\mu}^i \in [0, 1]$.

Let's denote the components of the TPMs by $\pi_{k|\mu}^i$:

$$\pi_{k|\mu}^i = P(\underline{x}_i = k \mid pa(\underline{x}_i) = \mu) \approx N_{k|\mu}^i . \quad (65.47)$$

The rest of this section lists equations that we obtained from the previous section, by adding the new indices i, μ :

$$\mathcal{K}(N_{\cdot,\mu}^i, \alpha_{\cdot,\mu}^i) = \frac{B(N_{\cdot,\mu}^i + \alpha_{\cdot,\mu}^i)}{CI(N_{\cdot,\mu}^i)B(\alpha_{\cdot,\mu}^i)} \quad (65.48)$$

$$\hat{\pi}_{k|\mu}^i = \frac{N_{k,\mu}^i + \alpha_{k,\mu}^i}{N_{+,\mu}^i + \alpha_{+,\mu}^i} \quad (65.49)$$

$$P(N_{\cdot,\mu}^i | \alpha_{\cdot,\mu}^i) = \mathcal{K}(N_{\cdot,\mu}^i, \alpha_{\cdot,\mu}^i) \quad (65.50)$$

$$P(N_{\cdot,\mu}^i | \pi_{\cdot|\mu}^i, \alpha_{\cdot,\mu}^i) \approx Mul(N_{\cdot,\mu}^i + \alpha_{\cdot,\mu}^i; \pi_{\cdot|\mu}^i, N_{+,\mu}^i + \alpha_{+,\mu}^i) \quad (65.51)$$

Claim 92

$$\ln P(N_{\cdot,\mu}^i | \hat{\pi}_{\cdot|\mu}^i, \alpha_{\cdot,\mu}^i) = \sum_k (N_{k,\mu}^i + \alpha_{k,\mu}^i) \ln \left(\frac{N_{k,\mu}^i + \alpha_{k,\mu}^i}{N_{+,\mu}^i + \alpha_{+,\mu}^i} \right) + \ln C(N_{\cdot,\mu}^i + \alpha_{\cdot,\mu}^i) \quad (65.52)$$

$$> \sum_k (N_{k,\mu}^i + \alpha_{k,\mu}^i) \ln \left(\frac{N_{k,\mu}^i + \alpha_{k,\mu}^i}{N_{+,\mu}^i + \alpha_{+,\mu}^i} \right) - \frac{1}{2}(nk^i - 1) \ln N_{+,\mu}^i \quad (65.53)$$

65.4 Bayesian Scores

- Bayesian Information Criterion (BIC)

$$\text{BIC-score} = - \sum_i \sum_{k,\mu} N_{k,\mu}^i \ln \left(\frac{N_{k,\mu}^i}{N_{+, \mu}^i} \right) + \underbrace{-\frac{|B|}{2} \ln N_{+,+}^+}_{\sum_i \sum_\mu \ln C(N_{\cdot, \mu}^i) \text{ would be more accurate}} \quad (65.54)$$

$$\approx \sum_i \sum_\mu \ln P(N_{\cdot, \mu}^i | \hat{\pi}_{\cdot, \mu}^i, \alpha_{\cdot, \mu}^i = 0) \quad (65.55)$$

- Bayesian Dirichlet (BD)

$$\text{BD-score} = \sum_i \sum_\mu \ln \frac{B(N_{\cdot, \mu}^i + \alpha_{\cdot, \mu}^i)}{B(N_{\cdot, \mu}^i)} \quad (65.56)$$

$$= \sum_i \sum_\mu \ln [CI(N_{\cdot, \mu}^i) P(N_{\cdot, \mu}^i | \alpha_{\cdot, \mu}^i)] \quad (65.57)$$

- BD equivalent (BDe)

$$\text{BDe-score} = \text{BD-score} (\alpha_{k,\mu}^i = \alpha' N_{k,\mu}^i) , \quad (65.58)$$

where α' is a free parameter.

- BD equivalent unified (BDeu)

$$\text{BDeu-score} = \text{BD-score} \left(\alpha_{k,\mu}^i = \frac{\alpha'}{n k^i n \mu^i} \right) , \quad (65.59)$$

where α' is a free parameter. The BDeu score satisfies **score equivalence**; i.e., it is the same for all DAGs in an equivalence class of observational equivalent DAGs. See Chapter 53 for more information about observational equivalence.

65.5 Information Theoretic scores

- Maximum likelihood

$$\text{ML-score} = \sum_i \sum_{k,\mu} N_{k,\mu}^i \ln N_{k|\mu}^i \quad (65.60)$$

$$= - \sum_i H(\underline{k}^i | \underline{\mu}^i) , \quad (65.61)$$

where $P_{\underline{k}^i | \underline{\mu}^i}(k | \mu) = N_{k|\mu}^i$ and $P_{\underline{k}^i, \underline{\mu}^i}(k, \mu) = N_{k,\mu}^i$.

- Bayesian Information Criterion (BIC), aka Minimum Description Length (MDL)

$$\text{BIC-score} = \text{ML-score} - \frac{|B|}{2} \ln N_{+,+}^+ \quad (65.62)$$

$$\approx \sum_i \sum_{k,\mu} N_{k,\mu}^i \ln \frac{N_{k|\mu}^i}{\sqrt{N_{+,+}^+}} \quad (65.63)$$

- Akaike Information Criterion (AIC)

$$\text{AIC-score} = \text{ML-score} - |B| \quad (65.64)$$

$$\approx \sum_i \sum_{k,\mu} N_{k,\mu}^i [\ln N_{k|\mu}^i - 1] \quad (65.65)$$

Chapter 66

Selection Bias Removal

This chapter is based on Ref.[1].

Selection bias (SB) occurs when one samples from an atypical subset of a population, producing a biased dataset. Are such biased datasets useless? Not necessarily. It is possible to add auxiliary features to the biased dataset, and to sample those new features in an unbiased way, from the whole population. Then one can merge the original biased dataset with the auxiliary, unbiased one, to obtain an enhanced dataset. It is sometimes possible to do this so that the enhanced dataset is provably unbiased. It's like making horizontal the surface of a table that was not initially horizontal. The theory of Bayesian Networks and Causal Inference tells us WHEN this is possible, and HOW to do it when it is possible.

Consider the bnet Fig.66.1.

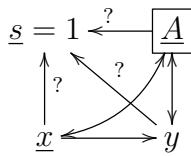


Figure 66.1: Bnet considered for selection bias (SB) removal. Arrows with question marks may or may not be present.

Let

$\underline{s} \in \{0, 1\}$ be the **selection node**. $\underline{s} = 1$ means there is SB in the parent nodes.

\underline{x} = **class features**.¹

\underline{y} = **target feature**.

\underline{A} = **auxiliary features**. This is a set of nodes that may contain arrows entering or exiting it, as indicated by the double arrows.

$\underline{E} = \{\underline{y}, \underline{x}\} \cup \underline{A}$ = **Enhanced feature set**.

¹A feature is the same as a node in a bnet.

Σ = unbiased population of individuals σ

Σ_o = biased sub-population of individuals, $\Sigma_o \subset \Sigma$.

$OD = \{(\sigma_o, \underline{x}^{\sigma_o}, \underline{y}^{\sigma_o}, \underline{s}^{\sigma_o} = 1) : \sigma_o \in \Sigma_o\}$ = **Original Dataset**, dataset for $(\underline{x}, \underline{y})$ features with $\underline{s} = 1$. Gives empirical distribution $P(\underline{y}|\underline{x}, \underline{s} = 1)$. (Ref.[1] calls this dataset the **biased study**.)

$AD = \{(\sigma, \underline{x}^\sigma, \underline{A}^\sigma) : \sigma \in \Sigma\}$ = **Auxiliary Dataset**, dataset for $(\underline{x}, \underline{A})$ features. Gives empirical distribution $P(\underline{A}|\underline{x})$. (Ref.[1] calls this dataset the **population level study**.)

$ED = \{(\sigma_o, \underline{x}^{\sigma_o}, \underline{A}^{\sigma_o}, \underline{y}^{\sigma_o}, \underline{s}^{\sigma_o} = 1) : \sigma_o \in \Sigma_o\}$ = **Enhanced Dataset**, dataset for $(\underline{x}, \underline{y}, \underline{A})$ features for $\underline{s} = 1$. Obtained by merging OD and AD . Gives empirical distribution $P(\underline{y}|\underline{x}, \underline{A}, \underline{s} = 1)$.

66.1 Pre and Post Selection Nodes

Selection nodes can be a source of confusion, so it's worth spending some time thinking about them.

This book uses 2 types of selection nodes: pre-selection nodes and post-selection nodes.

pre-selection nodes are used in Chapter 76 entitled “Transportability of Causal Knowledge”. Suppose that the nodes of a bnet can be either blue or colorless. Then we can define a pre-selection node \underline{s}_{blue} such that $\underline{s}_{blue} = 1$ points to the nodes that are blue, and nothing points to the nodes that are colorless. If we were to change the value of node \underline{s}_{blue} to zero, then all the nodes of the bnet would be colorless. Suppose instead that the nodes of a bnet could be either blue, red or colorless. Then one could define two pre-selection nodes, \underline{s}_{blue} and \underline{s}_{red} , and draw arrows from $\underline{s}_{blue} = 1$ to the blue nodes and from $\underline{s}_{red} = 1$ to the red nodes, and no arrow directed into the colorless nodes. So, basically, pre-selection nodes are binary **root** nodes that indicate whether the nodes pointed to belong to a set or not.

Most of the DAG literature, including Ref.[1], on which this chapter is based, define SB using a **post-selection node**. A post-selection node is a binary **leaf** node of the graph.

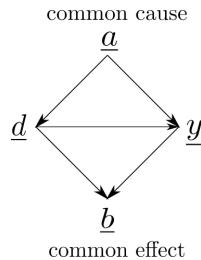


Figure 66.2: Common Cause and Effect for nodes $\underline{d}, \underline{y}$.

Note that in Potential Outcomes (PO) theory (see Chapter 56), pre-selection

nodes such as \underline{a} in Fig.66.2 are called **common cause (confounder, fork) nodes for nodes $\underline{d}, \underline{y}$** . Furthermore, post-selection nodes such as \underline{b} in Fig.66.2 are called **common effect (selection bias (SB), collider) nodes for nodes $\underline{d}, \underline{y}$** . Hence, in PO theory, SB is indicated by a leaf node, just as we do in this chapter.

Note that **Simpson's paradox** (see Chapter 69) arises from an indirect effect caused by not conditioning on a confounder, whereas **Berkson's paradox** (see Chapter 7) arises from an indirect effect caused by conditioning on a collider.

It's possible to replace a pre-selection node by a post selection node, or vice versa, as follows. Suppose that we start with a bnet G_0 that is fully connected, and we add to it a selection node \underline{s} that is a root node that points to all nodes of G_0 . Call the resulting bnet $\underline{s} \rightarrow G_0$. We can use Bayes rule to reverse the direction of the arrows emanating from \underline{s} so that they enter node \underline{s} rather than exit it. Call the resulting bnet $\underline{s} \leftarrow G_0$. In general, Bayes rule allows us to translate from $\underline{s} \rightarrow G_0$ to $\underline{s} \leftarrow G_0$, or in the opposite direction, without having to change the directions of any of the arrows in G_0 . If G_0 is not fully connected, then going from $\underline{s} \rightarrow G_0$ to $\underline{s} \leftarrow G'_0$ will often require that G'_0 have the same arrows in the same directions as G_0 plus some extra arrows new to G_0 . Likewise, going from $\underline{s} \leftarrow G_0$ to $\underline{s} \rightarrow G''_0$ may require that G''_0 have the same arrows as G_0 plus some new arrows.

So far, we have been intentionally vague in specifying the graphs G'_0 and G''_0 . In Fig.66.3 we give a trick for determining possible candidates for graphs G'_0 and G''_0 . In Fig.66.3, we consider 3 panels going from left to right, depicting the cases where \underline{s} has either 1,2 or 3 neighbors. The top graph $G_{post\text{-}sel}$, which has a post-selection node \underline{s} , is converted to a graph which is numerically equal to it, namely the bottom graph $G_{pre\text{-}sel}$, which has a pre-selection node \underline{s} . The magenta arrows represent any number of arrows exiting (but none entering) a node. If we start with a graph $\underline{s} \leftarrow G_0$, we find the biggest subset X of the nodes of G_0 such that $\underline{s} \leftarrow X$ only has nodes exiting it (i.e., only magenta nodes). Then we add enough arrows to $\underline{s} \leftarrow X$ to make it a fully connected graph $\underline{s} \leftarrow X'$. Now we can reverse the incoming arrows to \underline{s} and make them all outgoing and call the resulting graph $\underline{s} \rightarrow X'$.

Recall that in Chapter Definition of a Bayesian Network, we made a distinction between causally correct (CC) bnets, and non CC bnets, and we pointed out that non CC bnets are quite useful as a numerical tool. Recall also from Chapter 53 that two bnets can be “observationally equivalent”. That is what is happening here. We are faced with the choice of making selection nodes either leaf nodes or root nodes. Both choices lead to observationally equivalent bnets. One of the two choices leads to the CC bnet, and the other to a non-CC bnet. Both choices are numerically correct.

In this chapter, we will consider first a DAG $G_{post\text{-}sel}$ with a post-selection node, then we will transform that DAG to a numerically equal DAG $G_{pre\text{-}sel}$ with a pre-selection node. The latter DAG is more convenient for our needs. Why? Because in the SB theory that we present in this chapter, the $\underline{s} = 1$ appears as a condition in a conditional probability, as in $P(\dots | \dots, \underline{s} = 1)$. Such probabilities are represented in a more straightforward manner if arrows exit rather than enter node \underline{s} .

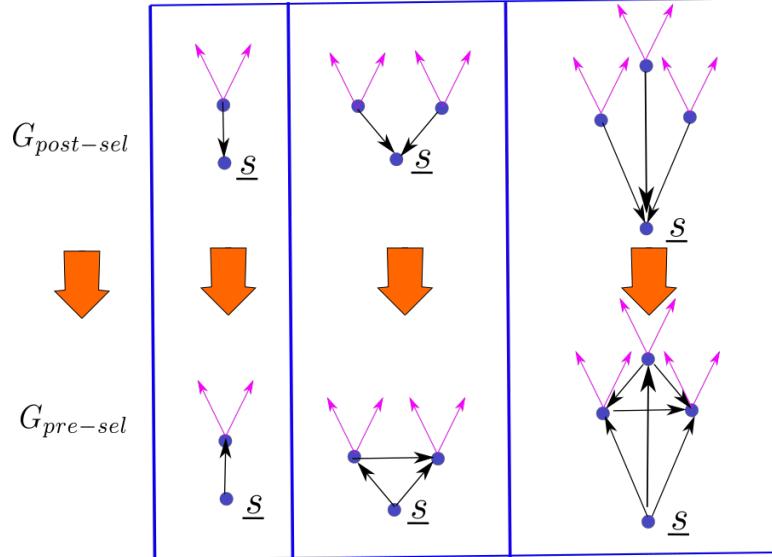


Figure 66.3: Switching from a post-selection node to a pre-selection node.

66.2 Removing SB from passive query $P(y|x)$

The **passive query** $Q = P(y|x, \underline{s} = 1)$ is **SB-recoverable** if it is independent of \underline{s} .

1. **Query $P(y|x)$ is SB-recoverable** with $\underline{A} = \emptyset$; SB can be removed by conditioning on \underline{x} .

If $\underline{y} \perp \underline{s}|\underline{x}$, then

$$P(y|x, \underline{s} = 1) = P(y|x). \quad (66.1)$$

For example,

$$\begin{array}{ccc} \underline{s} & & \underline{s} \\ \uparrow & & \downarrow \\ \underline{x} \longrightarrow \underline{y} & = & \underline{x} \longrightarrow \underline{y} \end{array} \quad (66.2a)$$

$$\begin{array}{ccc} \underline{s} \leftarrow \underline{a} & & \underline{s} \longrightarrow \underline{a} \\ \uparrow & & \downarrow \\ \underline{x} \longrightarrow \underline{y} & = & \underline{x} \longrightarrow \underline{y} \end{array} \quad (66.2b)$$

$$\begin{array}{ccc} \underline{s} \leftarrow \underline{a} & & \underline{s} \longrightarrow \underline{a} \\ \uparrow & & \downarrow \\ \underline{x} \longrightarrow \underline{y} & = & \underline{x} \longrightarrow \underline{y} \end{array} \quad (66.2c)$$

The bnets on the left hand sides of Eqs.(66.2) satisfy $y \perp \underline{s} | \underline{x}$.

2. **Query $P(y|x)$ is SB-recoverable** via \underline{a} ; SB can be removed by conditioning on \underline{x} and \underline{a} . Here \underline{a} is a nonempty subset of \underline{A}

Claim 93 *There exists $\underline{a} \subset \underline{A}$ such that $y \perp \underline{s} | (\underline{x}, \underline{a})$ and $\underline{a} \perp \underline{s} | \underline{x}$ iff*

$$P(y|x, \underline{s} = 1) = \sum_{\underline{a}} \underbrace{P(y|x, a, \underline{s} = 1)}_{P(y|x,a)} \underbrace{P(a|x, \underline{s} = 1)}_{P(a|x)} = P(y|x) \quad (66.3)$$

$$\begin{array}{c} \underline{s} = 1 \\ \searrow \quad \nearrow \\ x \longrightarrow y \end{array} = \begin{array}{c} \sum a \\ \nearrow \quad \searrow \\ x \longrightarrow y \end{array} = \begin{array}{c} x \longrightarrow y \end{array} \quad (66.4)$$

proof:

The \Rightarrow part of this claim is obvious. For a proof of the \Leftarrow part, see Ref.[1].
QED

For example,

$$\begin{array}{c} s \\ \uparrow \quad \downarrow \\ x \longrightarrow y \end{array} = \begin{array}{c} s \\ \downarrow \quad \downarrow \\ x \longrightarrow y \end{array} \quad (66.5a)$$

The bnets on the left hand sides of Eqs.(66.5) satisfy $y \perp \underline{s} | (x, a)$ and $a \perp \underline{s} | x$.

3. **Query $P(y|x)$ is not SB-recoverable;** SB cannot be removed.

For example,

$$\begin{array}{c} s \\ \swarrow \quad \searrow \\ x \longrightarrow y \end{array} = \begin{array}{c} s \\ \downarrow \quad \searrow \\ x \longrightarrow y \end{array} \quad (66.6)$$

$$\begin{array}{c} s \\ \uparrow \quad \searrow \\ x \longrightarrow y \end{array} = \begin{array}{c} s \\ \downarrow \quad \searrow \\ x \longrightarrow y \end{array} \quad (66.7)$$

$$\begin{array}{ccc}
 \begin{array}{c} \underline{s} \leftarrow \underline{a} \\ \uparrow \\ \underline{x} \xrightarrow{\quad} \underline{y} \\ \uparrow \\ \underline{w} \end{array} & = & \begin{array}{c} \underline{s} \longrightarrow \underline{a} \\ \downarrow \\ \underline{x} \xrightarrow{\quad} \underline{y} \\ \uparrow \\ \underline{w} \end{array}
 \end{array} \tag{66.8}$$

66.3 Removing SB from active query $P(y|\mathcal{D}\underline{x})$

The **active query (i.e., do query)** $Q = P(y|\mathcal{D}\underline{x} = \underline{x}, \underline{s} = 1)$ is

- (a) **SB-recoverable** if it equals $P(y|\mathcal{D}\underline{x} = \underline{x})$,
- (b) **do-identifiable** if it equals $P(y|\underline{x}, \underline{s} = 1)$.
- (c) both **SB-recoverable and do-identifiable** if it equals $P(y|\underline{x})$.

Examples

- $$\begin{array}{ccc}
 \begin{array}{c} \underline{s} \leftarrow \underline{a} \\ \downarrow \\ \underline{x} \xrightarrow{\quad} \underline{y} \end{array} & = & \begin{array}{c} \underline{s} \longrightarrow \underline{a} \\ \downarrow \\ \underline{x} \xrightarrow{\quad} \underline{y} \end{array}
 \end{array} \tag{66.9}$$

SB-recoverable: NO
do-identifiable: YES

$Q = P(y|\mathcal{D}\underline{x} = \underline{x}, \underline{s} = 1)$ is do-identifiable because the bnet contains no hidden variables. It's s-recoverable because the bnet on the left hand side of Eq.(66.9) satisfies $\underline{y} \perp \underline{s} | (\underline{x}, \underline{a})$ but Not $\underline{a} \perp \underline{s} | \underline{x}$.

- $$\begin{array}{ccc}
 \begin{array}{c} \underline{s} \\ \uparrow \\ \underline{x} \xrightarrow{\quad} \underline{y} \end{array} & = & \begin{array}{c} \underline{s} \\ \downarrow \\ \underline{x} \xrightarrow{\quad} \underline{y} \end{array}
 \end{array} \tag{66.10}$$

SB-recoverable: YES
do-identifiable: NO

Let

V = set of nodes in graph

$V^{<\underline{x}}$ = non-descendants of \underline{x} (excluding \underline{x})

$V^{>\underline{x}}$ = descendants of \underline{x} (excluding \underline{x})

$\underline{z}^{<\underline{x}} = \underline{z} \cap V^{<\underline{x}}$

$\underline{z}^{>\underline{x}} = \underline{z} \cap V^{>\underline{x}}$

Suppose $\underline{z} \cup \{\underline{x}, \underline{y}\} \subset E$ and $\underline{z} \subset \underline{A}$. We say \underline{z} satisfies the **selection bias (SB) backdoor criterion** with respect to $(\underline{x}, \underline{y})$ if

1. all backdoor paths from \underline{x} to \underline{y} are blocked by conditioning on $\underline{z}^{<\underline{x}}$
2. $\underline{z}^{>\underline{x}} \perp \underline{y} | (\underline{x}, \underline{z}^{<\underline{x}})$
3. $\underline{y} \perp \underline{s} | (\underline{x}, \underline{z})$

Claim 94 (*SB Backdoor Adjustment Formula*)

If \underline{z} satisfies the SB backdoor criterion relative to $(\underline{x}, \underline{y})$, then

$$P(y | \mathcal{D}\underline{x} = x, \underline{s} = 1) = \sum_z P(y|x, z)P(z) = P(y|x) \quad (66.11)$$

$$\begin{array}{c} \underline{s} = 1 \\ \searrow \\ \mathcal{D}\underline{x} = x \longrightarrow y \end{array} = \begin{array}{c} \sum z \\ \downarrow \\ x \longrightarrow y \end{array} = x \longrightarrow y \quad (66.12)$$

proof:

If z satisfies the SB backdoor criterion relative to $(\underline{x}, \underline{y})$, then $\underline{x}, \underline{y}, \underline{z}$ might have the following structure.

$$\begin{array}{c} \underline{s} \xleftarrow{} \underline{z}^{<\underline{x}} \xrightarrow{} \underline{z}^{>\underline{x}} \\ \uparrow \quad \downarrow \\ \underline{x} \longrightarrow \underline{y} \end{array} = \begin{array}{c} \underline{s} \xrightarrow{} \underline{z}^{<\underline{x}} \xrightarrow{} \underline{z}^{>\underline{x}} \\ \downarrow \quad \downarrow \\ \underline{x} \longrightarrow \underline{y} \end{array} \quad (66.13)$$

See Claim 34 for a proof of this claim for the special case Eq.(66.13).

QED

Chapter 67

Sequential Backdoor Adjustment Formula



Figure 67.1: Piscina Mirabilis, ancient Roman cistern in Naples

This chapter is based on Ref.[44] by Pearl and Robins.

The goal of this chapter is to generalize the backdoor adjustment formula (see Chapter 3) from a query $P(y|\mathcal{D}\underline{x} = \underline{x})$ with a single do node to a query $P(y|\mathcal{D}\underline{x}^n = \underline{x}^n)$ with multiple do nodes.

For $n = 1, 2, 3 \dots$, define

$$\mathcal{Q}(y|x^n) = \sum_{z^n} P(y|x^n, z^n) \prod_{t=0}^{n-1} P(z_t|x_{<t}, z_{<t}) \quad (67.1)$$

$$= \begin{array}{c} \sum_{z^n} z^n \\ \downarrow y \\ \prod_{t=0}^{n-1} \\ \uparrow \\ x^n \end{array} \quad \begin{array}{c} z_{<t} \longrightarrow z_t \\ \nearrow \\ x_{<t} \end{array} \quad (67.2)$$

For $n = 1$,

$$\mathcal{Q}(y|x_0) = \begin{array}{c} \sum_{z_0} z_0 \\ \downarrow y \\ \uparrow \\ x_0 \end{array} = \begin{array}{c} \sum z_0 \\ \searrow \\ x_0 \end{array} \longrightarrow y \quad . \quad (67.3)$$

For $n = 2$,

$$\mathcal{Q}(y|x^2) = \begin{array}{c} \sum_{z^2} (z_0, z_1) \\ \downarrow y \\ \uparrow \\ (x_0, x_1) \end{array} \quad \begin{array}{c} P(z_0) \\ z_0 \longrightarrow z_1 \\ \nearrow \\ x_0 \end{array} \quad (67.4)$$

$$= \begin{array}{c} \sum z_0 \longrightarrow \sum z_1 \\ \nearrow \\ x_0 \end{array} \quad \begin{array}{c} \nearrow \\ x_1 \end{array} \quad \begin{array}{c} \longrightarrow \\ y \end{array} \quad . \quad (67.5)$$

For $n = 3$,

$$\mathcal{Q}(y|x^3) = \sum_{z^3} (z_0, z_1, z_2) P(z_0) z_0 \longrightarrow z_1 (z_0, z_1) \longrightarrow z_2 \quad (67.6)$$

$$= \sum_{z^3} (z_0, z_1, z_2) P(z_0) z_0 \overrightarrow{\longrightarrow} z_1 \overrightarrow{\longrightarrow} z_2 \quad (67.7)$$

$$= \sum z_0 \overrightarrow{\longrightarrow} \sum z_1 \overrightarrow{\longrightarrow} \sum z_2 \quad (67.8)$$

Suppose that we have access to data that allows us to estimate a probability distribution $P(x^n, y, z^n)$. Hence, the variables $\underline{x}^n, \underline{y}, \underline{z}^n$ are ALL observed (i.e, not hidden). Then we say that the the multinode of “covariates” \underline{z}^n satisfies the **sequential backdoor adjustment criterion** relative to $(\underline{x}^n, \underline{y})$ if for all $t \in \{0, 1, \dots, n-1\}$,

1. $\underline{y} \perp \underline{x}_t \mid \underbrace{(\underline{x}_0, \underline{x}_1, \dots, \underline{x}_{t-1}, \underline{z}_0, \underline{z}_1, \dots, \underline{z}_t)}_{\text{Past of } \underline{x}_t} \text{ in } \mathcal{L}_{\underline{x}_t} \mathcal{D}_{\underline{x}_{t+1}, \underline{x}_{t+2}, \dots, \underline{x}_{n-1}} G.$
2. $\underline{z}_t \cap de(\underline{x}_t) = \emptyset$.

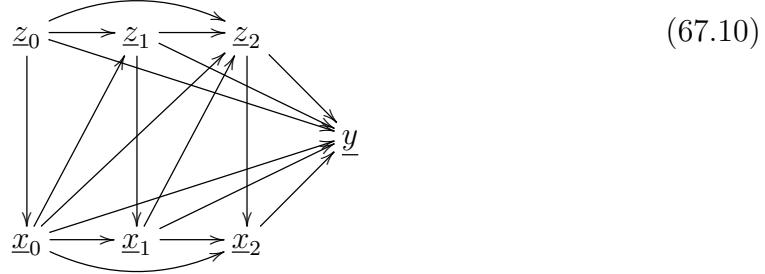
Claim 95 (*Sequential Backdoor Adjustment Formula*)

If \underline{z}^n satisfies the sequential backdoor criterion relative to $(\underline{x}^n, \underline{y})$, then

$$P(y | \mathcal{D}\underline{x}^n = x^n) = \mathcal{Q}(y|x^n), \quad (67.9)$$

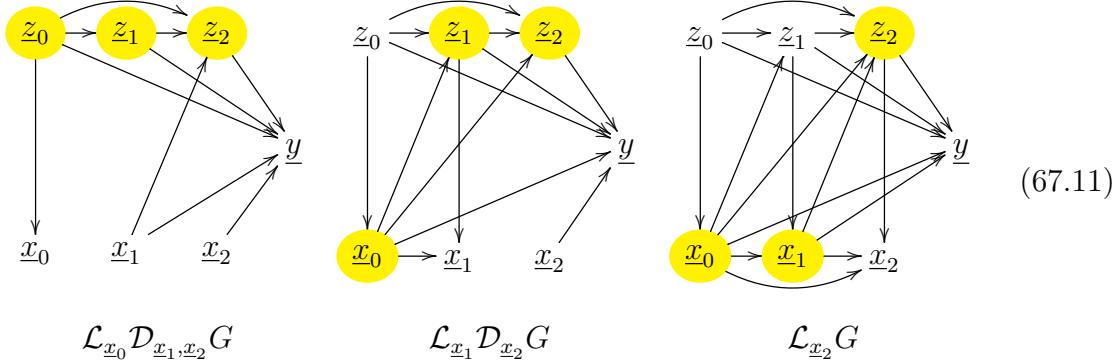
where $\mathcal{Q}(y|x^n)$ is defined by Eq.(67.2).

proof: If z^n satisfies the sequential backdoor criterion relative to $(\underline{x}^n, \underline{y})$, then $\underline{x}^n, \underline{y}, \underline{z}^n$ might have the following structure for $n = 3$.



(67.10)

One can check using the following 3 auxiliary bnets that bnet Eq.(67.10) satisfies the sequential backdoor criterion. Note that conditioned nodes are shaded yellow.



$\mathcal{L}_{\underline{x}_0} \mathcal{D}_{\underline{x}_1, \underline{x}_2} G$

$\mathcal{L}_{\underline{x}_1} \mathcal{D}_{\underline{x}_2} G$

$\mathcal{L}_{\underline{x}_2} G$

(67.11)

See Claim 33 for a proof of this claim for the special case Eq.(67.10).

QED

Chapter 68

Shapley Explainability

This chapter is based on Refs.[25] and [26], which I highly recommended.

“AI” is an ill-defined term. So is the term “explainability”. So the term “Explainable AI (XAI)” is doubly ill-defined. In 2018, the European Union codified the need for XAI — because of an individual’s “right to explanation” — into a law called the General Data Protection Right (GDPR). This EU law was a strong motivation for Neural Net and boosted decision tree practitioners to come up with a way to enhance their machine learning algorithms so that these comply with that law. Shapley explainability (SX) is one of the most popular methods for doing XAI.

So what does SX do? It ranks, for each individual of a population, the features (for example, race) of a dataset, in the order of how influential those features were in arriving at the decision the classifier made for that individual.

In my opinion, the goal of XAI is accomplished better with bnets than with SX enhanced NNs. SX “explains”, *a posteriori*, the outcome of a model, whereas bnets reveal the *a priori* process whereby that outcome was reached. Thus, SX can tell you that a model is racist, but it can’t suggest how to fix it. On the other hand, if a bnet is acting racist, you don’t have to throw it away. It can be fixed. Another weakness of SX is that it is quite expensive computationally. Bnets have explainability built into them. For bnets, explainability is not an additional, posterior and quite onerous, calculation. That is why I like to call bnets the gold standard of XAI.

Let

F be the feature set. For example, $F = \{age, gender, job\}$.

$\mathcal{P}(A) = \{S : S \subset A\}$ be the power set of the set A (*i.e.*, the set of all subsets of A , including the empty set \emptyset).

$$|\mathcal{P}(A)| = \sum_{k=0}^{|A|} \binom{|A|}{k} = \sum_{k=0}^{|A|} \binom{|A|}{k} 1^k 1^{|A|-k} = (1+1)^{|A|} = 2^{|A|}$$

$\mathcal{P}_f(F) = \{S \in \mathcal{P}(F) : f \in S\}$ for $f \in F$, be all sets in $\mathcal{P}(F)$ containing feature f . Note that $\mathcal{P}_f(F) = \mathcal{P}(F) - \mathcal{P}(F - \{f\})$

$\mathcal{P}_{!f}(F) = \{S \in \mathcal{P}(F) : f \notin S\}$ for $f \in F$, be all sets in $\mathcal{P}(F)$ not containing feature f . Note that $\mathcal{P}_{!f}(F) = \mathcal{P}(F - \{f\})$

Fig.68.1 shows a graph of $\mathcal{P}(F)$ for $F = \{age, gender, job\}$. Henceforth, we will refer to the generalization of Fig.68.1 to an arbitrary finite set F , as the **power**

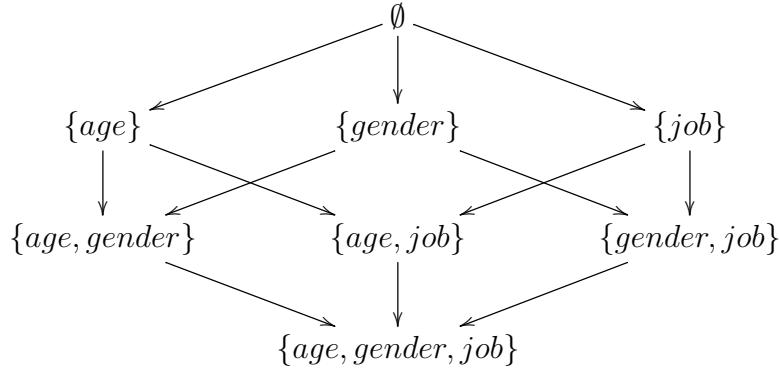


Figure 68.1: Graph of $\mathcal{P}(F)$ for $F = \{\text{age}, \text{gender}, \text{job}\}$. An arrow $H \leftarrow T$, where $H \in \mathcal{P}(F)$ is the head of the arrow and $T \in \mathcal{P}(F)$ is the tail of the arrow, means $H \supset T$ and $|H| = |T| + 1$.

set graph of F .¹

For any finite set F , consider its power set graph. Let $S \in \mathcal{P}(F)$ be a node of that graph. Let

$$N_{arr/nd}(S) = |S| = \text{number of arrows entering node } S.$$

$$N_{nds}(|S|) = \binom{|F|}{|S|} = \text{number of nodes in level } |S| \text{ (i.e., row } |S|).$$

$N_{arr}(S) = N_{arr/nd}(S)N_{nds}(|S|) = |S| \binom{|F|}{|S|} = \text{number of arrows going from row } |S| - 1 \text{ to row } |S|.$

$$P(S) = \frac{1}{N_{arr}(S)} = \frac{1}{|S| \binom{|F|}{|S|}} = \frac{1}{|F| \binom{|F|-1}{|S|-1}}. \quad (68.1)$$

Claim 96

$$\sum_{S \in \mathcal{P}_f(F)} P(S) = 1 \quad (68.2)$$

proof:

¹Note that a power set graph is a DAG. We won't define TPMs for its nodes in this chapter, so it's a DAG but not a bnet, in this chapter at least.

$$\sum_{S \in \mathcal{P}_f(F)} P(S) = \sum_{S \in \mathcal{P}_f(F)} \frac{1}{|F| \binom{|F|-1}{|S|-1}} \quad (68.3)$$

$$= \sum_{k=1}^{|F|} \sum_{S \in \mathcal{P}_f(F)} \mathbb{1}(|S|=k) \frac{1}{|F| \binom{|F|-1}{k-1}} \quad (68.4)$$

$$= \sum_{k=1}^{|F|} \frac{1}{|F| \binom{|F|-1}{k-1}} \underbrace{\sum_{S \in \mathcal{P}_f(F)} \mathbb{1}(|S|=k)}_{\binom{|F|-1}{k-1}} \quad (68.5)$$

$$= 1 \quad (68.6)$$

QED

Consider any $S \in \mathcal{P}(F)$. Henceforth, we will represent a Machine Learning (ML) model ML_S as follows. ML_S can be a Linear Regression (LR_S) model, a Neural Net model (NN_S), or any other type of ML model. We will list a dataset; i.e., a set of tuples indexed by the individuals σ of a population Σ such that $|\Sigma| = nsam$. The independent variables of ML_S (i.e., $x_S^\sigma = [x_f^\sigma : f \in S]$) will be shown unboxed and the dependent variable (aka target feature) (i.e., y^σ) will be shown inside a box. Then we will show an arrow with the superscript “ML-fit”, followed by the fit function obtained by performing ML_S .

$$ML_S : \{(\sigma, x_S^\sigma, \boxed{y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \hat{y}(x_S) \quad (68.7)$$

For any feature $f \in F$ and individual $\sigma \in \Sigma$, define the **Shapley Value** (SHAP) by

$$SHAP_f^\sigma = \sum_{S \in \mathcal{P}_f(F)} P(S) [\hat{y}(x_S^\sigma) - \hat{y}(x_{S-\{f\}}^\sigma)] \quad (68.8)$$

$$= E_S[\hat{y}(x_S^\sigma) - \hat{y}(x_{S-\{f\}}^\sigma)] \quad (68.9)$$

Hence,

- $SHAP_f^\sigma$ is an average over the ensemble $\{ML_S : S \in \mathcal{P}_f(F)\}$ for each individual $\sigma \in \Sigma$.
- $SHAP_f^\sigma$ averages the change in output \hat{y} when we change the model from one without feature f to one with feature f .
- $SHAP_f^\sigma$ can be negative or positive. Zero $SHAP_f^\sigma$ for individual σ means feature f does not influence how the decision \hat{y} was arrived at for individual σ .

- An exact calculation of $SHAP_f^\sigma$, for all f and for a single σ , requires training a different ML model for each node of the power set graph of F , so it requires training $2^{|F|}$ models. Yikes! As $|F|$ grows, this quickly becomes unfeasible. For large $|F|$, one must resort to using sampling and approximations to get an approximation of the SHAP.

68.0.1 Numerical examples of SHAP

Next we present 2 numerical examples of SHAP. The figures and numerical values in this section were taken directly from Refs. [25] and [26].

1. Predicting Income from $F = \{age, gender, job\}$.

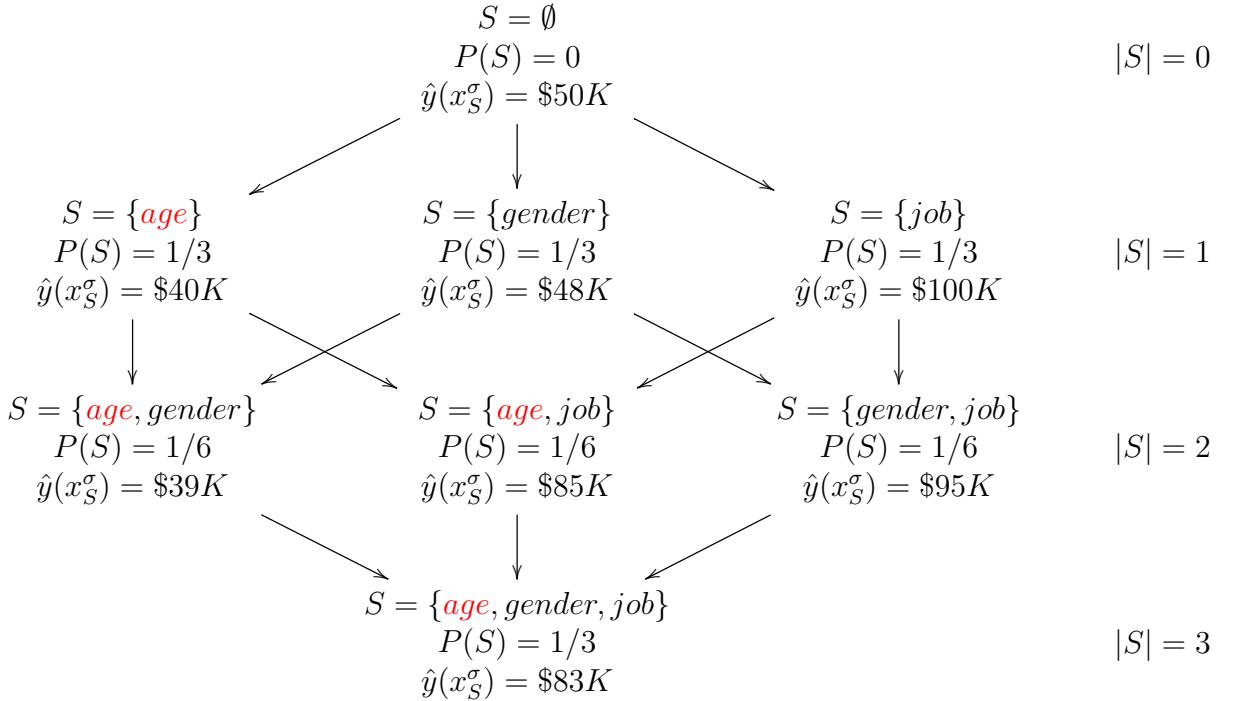


Figure 68.2: Same as Fig.68.1, but with added information for a specific individual σ . This figure contains enough information to evaluate $SHAP_{age}^\sigma$.

Consider the problem of predicting the income of a person based on the feature set $F = \{age, gender, job\}$. Suppose we are given a dataset for this problem. We can train models ML_S for each $S \in \mathcal{P}_{age}(F)$ where \hat{y} is the income. Then we can calculate the matrix $SHAP_{age}^\sigma$. Fig.68.2 gives all the information necessary to calculate $SHAP_{age}^\sigma$ for a single individual σ .

$$P(S) = \frac{1}{3^{\binom{|S|}{2}}} = \begin{cases} \frac{1}{3^{\binom{2}{0}}} = \frac{1}{3} & \text{if } |S| = 1 \\ \frac{1}{3^{\binom{2}{1}}} = \frac{1}{6} & \text{if } |S| = 2 \\ \frac{1}{3^{\binom{2}{2}}} = \frac{1}{3} & \text{if } |S| = 3 \end{cases} \quad (68.10)$$

$$SHAP_{age}^\sigma = \underbrace{P(age)}_{1/3} \underbrace{[\hat{y}(x_{age}^\sigma) - \hat{y}(x_\emptyset^\sigma)]}_{40K-50K} \quad (68.11)$$

$$+ \underbrace{P(age, job)}_{1/6} \underbrace{[\hat{y}(x_{age, job}^\sigma) - \hat{y}(x_{job}^\sigma)]}_{85K-100K} \quad (68.12)$$

$$+ \underbrace{P(age, gender)}_{1/6} \underbrace{[\hat{y}(x_{age, gender}^\sigma) - \hat{y}(x_{gender}^\sigma)]}_{39K-48K} \quad (68.13)$$

$$+ \underbrace{P(age, gender, job)}_{1/3} \underbrace{[\hat{y}(x_{age, gender, job}^\sigma) - \hat{y}(x_{gender, job}^\sigma)]}_{83K-95K} \quad (68.14)$$

$$= -\$11.33K \quad (68.15)$$

2. Predicting passenger survival in the Titanic disaster.

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
PassengerId							
1	3	male	22.0	1	0	7.2500	S
2	1	female	38.0	1	0	71.2833	C
3	3	female	26.0	0	0	7.9250	S
4	1	female	35.0	1	0	53.1000	S
5	3	male	35.0	0	0	8.0500	S

Figure 68.3: First five rows (passengers) of an abridged version of the Titanic Dataset available at kaggle.com. This figure shows (σ, x_F^σ) for $\sigma = 1, 2, \dots, 5$. It doesn't show the column $y^\sigma \in \{died, survived\}$.

Consider the problem of predicting whether an individual will survive or not based on a Titanic Dataset. We can train models ML_S for each $S \in \mathcal{P}(F)$ where $\hat{y} \in \{died, survived\}$. Then we can calculate the matrix $SHAP_f^\sigma$ for all individuals σ and features f . Fig.68.0.1 shows the first 5 rows of an abridged² version of the Titanic Dataset available at kaggle.com. Fig.68.4 displays $SHAP_f^\sigma$ in tabular form and Fig.68.5 displays $SHAP_f^\sigma$ in graphical form (in what is called a beeswarm plot).

²The Titanic Dataset available at kaggle.com has 891 rows and 15 columns, including columns for passenger ID and for $y^\sigma = survived? \in \{0, 1\}$. This abridged version has 8 columns.

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
PassengerId							
1	-0.36	-0.76	0.11	0.05	-0.03	-0.3	-0.08
2	1.22	2.18	0.1	0.06	0.1	0.78	0.42
3	-0.75	1.62	0.07	0.1	-0.02	-0.01	-0.13
4	1.15	2.07	0.11	0.05	0.08	0.89	-0.13
5	-0.41	-0.73	-0.06	0.08	-0.03	-0.15	-0.08

Figure 68.4: For the Titanic dataset, this is a table of $SHAP_f^\sigma$, where $\sigma \in \{1, 2, \dots, 5\}$ and $f \in F$. Cells with positive SHAP are colored green, and those with negative SHAP are colored red. The colors are not an indication of whether the passenger died or survived. Note that the table of a dataset and the matrix $SHAP_f^\sigma$ have the same shape ($|\Sigma|, |F|$).

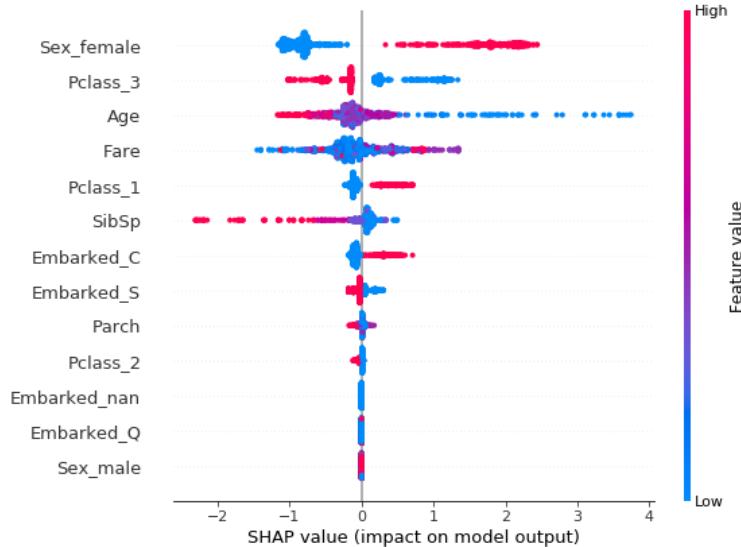


Figure 68.5: For the Titanic Dataset, this is a so called “beeswarm” plot of $SHAP_f^\sigma$, where $\sigma \in \{1, 2, \dots, 891\}$ and $f \in F$. In a beeswarm plot, the thickness of each row is proportional to how many individuals of the population have that value of the x coordinate. This plot comes from Ref.[25], where it was generated using the Titanic Dataset from kaggle.com and the wonderful Python library “SHAP”. The SHAP library can plot Shapley Values in many other styles besides this one.

Chapter 69

Simpson's Paradox

This chapter is based on Chapter 6 of “The Book of Why”, Ref.[43]. See also Ref.[125] and references therein.

Simpson’s paradox is a recurring nightmare for all statisticians overseeing a clinical trial for a medicine. It is possible that if they leave out a certain ”confounding” variable from a study, the study’s conclusion on whether a medicine is effective or not, might be, without measuring that confounding variable, the opposite of what it would have been had that variable been measured.

Simpson’s Paradox is greatly clarified by Judea Pearl’s theory of causality. At the end of this chapter, we explain how.

Here is a simple example of Simpson’s Paradox.

An equal number of patients of male and female genders are given a heart medicine or a placebo in a double blind study. Some subsequently have a heart attack. Let

\underline{a} = heart attack? No=0, Yes=1

\underline{t} = took medicine? No=0, Yes=1

\underline{g} = gender? Female=0, Male=1

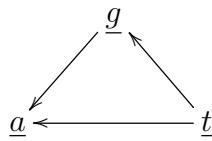


Figure 69.1: bnet for a simple example of Simpson’s paradox. Here node \underline{g} is a chain junction and a mediator.

This situation can be modeled by either bnet Fig.69.1. or bnet Fig.69.2. The two bnets are probabilistically equivalent (i.e., they both represent the same probability distribution $P(a, t, \underline{g})$) because

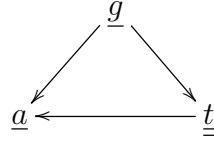


Figure 69.2: bnet that is probabilistically but not physically equivalent to bnet Fig.69.1. Here node \underline{g} is a fork junction and a confounder.

$$P(g|t)P(t) = P(g,t) = P(t|g)P(g) . \quad (69.1)$$

For the bnet Fig.69.1, one has

$$P(a,g,t) = P(a|g,t)P(g|t)P(t) . \quad (69.2)$$

Therefore,

$$P(a = 1|t) = \sum_g P(a = 1|t, g)P(g|t) = E_{\underline{g}|t}P(a = 1|t, \underline{g}) , \quad (69.3)$$

where $E_{\underline{g}|t}$ is a conditional expected value (a kind of weighted average).

Suppose q_0, q_1 are non-negative real numbers. For the vector $\vec{q} = (q_0, q_1)$:

Define a negative outcome (or failure or q_t increasing with t) if $q_0 \leq q_1$.

Define a positive outcome (or success or q_t decreasing with t) if $q_0 \geq q_1$.

Let

$$\vec{q}^g = [P(\underline{a} = 1|t, g)]_{t=0,1} \quad (69.4)$$

for $g = 0, 1$, and

$$\vec{q}^* = [P(\underline{a} = 1|t)]_{t=0,1} . \quad (69.5)$$

It is possible (see Fig.69.3 for a graphical explanation of how) to find perverse cases in which $P(a = 1|t, g = 0)$ and $P(a = 1|t, g = 1)$ increase with t but $P(a = 1|t)$ decreases with t . So it is possible to conclude that the medicine is a failure for each of the two g populations considered separately, yet the medicine is a success when both populations are “amalgamated”. The lesson is that a “trend reversal” is possible upon amalgamation. Trends are not necessarily preserved when we do a weighted average of type $E_{\underline{g}|t}$. $E_{\underline{g}|t}$ is an expected value on the random variable \underline{g} conditioned on the root random variable \underline{t} .

So far, we have proven that probabilistically, the drug can be a failure for the populations of both sexes considered separately, but a success for the aggregate population.

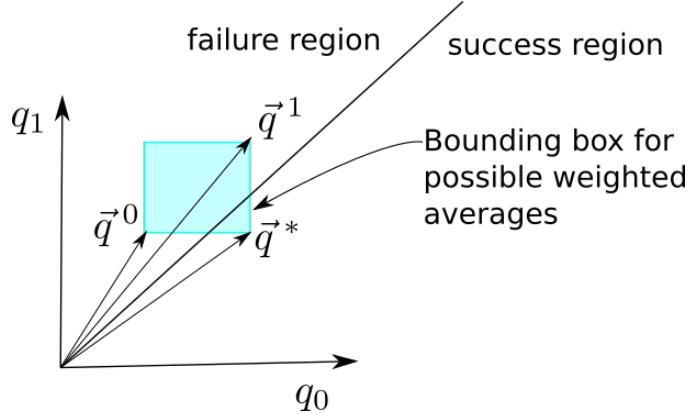


Figure 69.3: \vec{q}^0 , \vec{q}^1 vectors and bounding box for vector \vec{q}^* .

69.1 Pearl Causality

Pearl Causality would add the following two important insights to this problem:

1. bnets Fig.69.1 and Fig.69.2, although they are probabilistically equivalent, do not represent the same physical situation. In fact, only Fig.69.2 occurs in this case.
2. To decide whether the medicine is effective, we must apply a $do()$ operator to the \underline{t} variable in Fig.69.2. The effect of that $do()$ operator is to erase the arrow going from \underline{g} to \underline{t} . This in turn means that the average $E_{\underline{g}|\underline{t}}$ in our equation for $P(a = 1|\underline{t})$ becomes a simpler average $E_{\underline{g}}$ which is independent of \underline{t} . But for such an average, the bounding box in Fig.69.3 degenerates to its diagonal line that connects the tips of the two vectors \vec{q}^0 and \vec{q}^1 . The vector \vec{q}^* must now fall on that diagonal line and must therefore also fall in the success region.

In conclusion, as Judea Pearl would say, if we ask the right question to Nature, i.e., what is $P[a = 1|do(\underline{t} = t)]$ for $t = 0, 1$, we get as an answer that the aggregate population preserves rather than reverses the unanimous trend of the two gendered populations.

69.2 Numerical Example

(a, t, g)	number of patients segregated by gender	number of patients of either gender
0,0,0	19	47
0,0,1	28	
0,1,0	37	49
0,1,1	12	
1,0,0	1	13
1,0,1	12	
1,1,0	3	11
1,1,1	8	

Table 69.1: Data for numerical example of Simpson’s Paradox. This fictitious data was taken directly from Table 6.4, page 210 of “The Book of Why”, Ref.[43].

$$P(a|t, g) = \begin{array}{c|cccc} & 0,0 & 0,1 & 1,0 & 1,1 \\ \hline 0 & 19/20 & 28/40 & 37/40 & 12/20 \\ 1 & 1/20 & 12/40 & 3/40 & 8/20 \end{array} \quad (69.6)$$

$$P(a|t) = \begin{array}{c|cc} & 0 & 1 \\ \hline 0 & 47/60 & 49/60 \\ 1 & 13/60 & 11/60 \end{array} \quad (69.7)$$

$$\begin{aligned} \frac{P(a=1,t=1,g=0)}{\sum_a P(a,t=1,g=0)} &= P(a = 1|t = 1, g = 0) &= \frac{3}{40} \\ \frac{P(a=1,t=0,g=0)}{\sum_a P(a,t=0,g=0)} &= P(a = 1|t = 0, g = 0) &= \frac{1}{20} = \frac{2}{40} \end{aligned} \quad (69.8)$$

$$\begin{aligned} \frac{P(a=1,t=1,g=1)}{\sum_a P(a,t=1,g=1)} &= P(a = 1|t = 1, g = 1) &= \frac{8}{20} = \frac{16}{40} \\ \frac{P(a=1,t=0,g=1)}{\sum_a P(a,t=0,g=1)} &= P(a = 1|t = 0, g = 1) &= \frac{12}{40} \end{aligned} \quad (69.9)$$

$$\begin{aligned} \frac{\sum_g P(a=1,t=1,g)}{\sum_g \sum_a P(a,t=1,g)} &= P(a = 1|t = 1) &= \frac{11}{60} \\ \frac{\sum_g P(a=1,t=0,g)}{\sum_g \sum_a P(a,t=0,g)} &= P(a = 1|t = 0) &= \frac{13}{60} \end{aligned} \quad (69.10)$$

Note that the right hand side of Eq.(69.8) is higher for $t = 1$ than for $t = 0$. Same trend occurs in Eqs.69.9 but is reversed in Eqs.69.10.

Chapter 70

Structure and Parameter Learning for Bnets

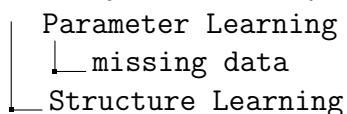
Learning a bnet from data is a computationally intensive NP-complete problem. Therefore, the best one can hope for is for heuristic algorithms that solve this problem approximately. A huge number of such algorithms have been tried and continue to be tried. Luckily, there exists a free open source software library called `bnlearn` that covers many of them. The goal of this chapter is to give a brief overview of the subject of bnet learning, after which we recommend to those readers who want to pursue this subject further, to learn `bnlearn`.

This chapter is based on the `bnlearn` website Ref.[49], and on a 2019 survey paper [50] by Scutari et al. I highly recommend looking at both. Refs. [3] and [23] were also helpful to me in understanding this subject.

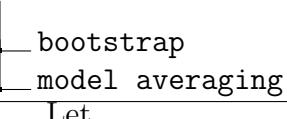
`bnlearn` (Ref.[49]) (free, open source) is very comprehensive and well maintained. It is written mostly in C with an R front-end. It was developed by Marco Scutari and collaborators over a time period of more than 10 years, and is still under active development. How things stand in the field of bnet learning software reminds me of how things stand in the field of linear algebra (LA) software. Perfecting and optimizing LA software takes many years so I would not advise you to write your own LA software library starting from scratch. There is no need to do so. Instead, you can use LAPACK (free, open source), which has been perfected and expanded for decades by world experts. I view `bnlearn` as the LAPACK of bnet learning.

70.1 Overview

To give the reader an overview of the subject and of `bnlearn` itself, here is a highly simplified tree, compiled from the `bnlearn` website and documentation, of some of the subjects covered by `bnlearn`.



```
tree-like structures given a priori
  Naive Bayes
  Chow-Liu tree
  Tree Augmented Naive Bayes (TAN)
  ARACNE
score based
algorithms
  hill climbing (HC)
  HC with random restarts
  HC with Tabu list (Tabu)
  simulated annealing
  genetic algorithms
scoring functions
  Information Theoretic scores
  Bayesian Information Criterion (BIC)
  Bayesian Dirichlet (BD) family
constraint based
algorithms
  PC family
  Grow-Shrink (GS)
  Incremental Association Markov Blanket (IAMB) family
conditional independence tests
  mutual information (parametric, semiparametric and permutation
    tests)
  shrinkage-estimator for the mutual information
hybrid
  Max-Min Hill Climbing (MMHC)
  Hybrid HPC (H2PC)
  General 2-Phase Restricted Maximization (RSMAX2)
parallel mode structure learning
node types
  all-discrete
  all-continuous
  mixed
utility functions
  model comparison and manipulation
  random data generation
  arc orientation testing
  simple and advanced plots
  parameter estimation (maximum likelihood and Bayesian)
  inference, conditional probability queries
  cross-validation
```



-
- Let
- PL=parameters learning (i.e, learning the TPMs)
 - SL= structure learning (i.e., learning the DAG)
 - ML= model (or bnet) learning, SL+PL

PL is easy, once the structure is known. PL assuming no missing data goes as follows. Using the notation of Chapter 65, define

$$\pi_{k|\mu}^i = P(\underline{x}_i = k \mid pa(\underline{x}_i) = \mu). \quad (70.1)$$

Then $\pi_{k|\mu}^i$ can be estimated from the data $N_{k,\mu}^i$ using:

$$\pi_{k|\mu}^i \approx N_{k|\mu}^i = \frac{N_{k,\mu}^i}{N_{+, \mu}^i}. \quad (70.2)$$

PL described by Eq.(70.2) is only for discrete nodes with no missing data. **bnlearn** can also do PL with missing data and continuous (Gaussian linear only) nodes. See Chapter 46 on missing data and Chapter 26 on Gaussian linear nodes. SL actually does PL and SL at the same time.

There are 3 main types of SL: score based, constraint based, and hybrid. **bnlearn** can perform many algorithms of each of these 3 types of SL. It can perform most of them with either all-discrete, or all-continuous or mixed nodes. It can perform many of them in parallel mode. The 2019 survey paper Ref.[50] by Scutari et al compares the performance of many different bnet learning algorithms.

70.2 Score based SL algorithms

Score based SL algorithms require scoring bnets (with either all-discrete, all-continuous or mixed nodes). See Chapter 65 for an introduction to scoring bnets. The BIC score explained in that chapter is very popular and works for all-discrete, all-continuous or mixed nodes.

Score-based SL algorithms apply standard optimisation techniques. In the Hill Climbing algorithm, the current best bnet is changed slightly and then given a score that measures how well it fits the data. The bnet with the highest (=best) score so far, as well as that highest score, are stored. (Hence, this is called a greedy search). The process continues until the latest highest score stops changing. The problem with being greedy all the time is that the answer might converge to a local maximum. To mitigate this problem and allow some probability of visiting more than one local maximum, one uses a Tabu Table, random restarts, simulated annealing, genetic algorithms, etc.

70.3 Constraint based SL algorithms

To fully understand constraint based SL algorithms, the reader is advised to read Chapters 19 and 53 first.

Constraint based SL algorithms require estimating from the data the conditional independence $\underline{x} \perp_P \underline{y} | \underline{a}$. for any 3 disjoint multinodes $\underline{x}, \underline{y}, \underline{a}$. This can be done by estimating the conditional mutual information (CMI) $H(\underline{x} : \underline{y} | \underline{a})$. `bnlearn` can calculate CMI and other metrics of $\underline{x} \perp_P \underline{y} | \underline{a}$. All these metrics are very similar; they all measure how close $P(x.|y., a.)$ and $\bar{P}(x.|a.)$ are.

The first constraint-based SL algorithm was the Inductive Causation (IC) algorithm proposed by Pearl and Verma in 1991. Incremental improvements have been proposed since then, such as the PC family of algorithms, Grow-Shrink and the Incremental Association Markov Blanket (IAMB) family of algorithms.

70.4 Pseudo-code for some bnet learning algorithms

Algorithm 2: Pseudo-code for Hill Climbing algorithm

Input : Data D , Vertices V

Output: a bnet $B = (G, T)$, where $G = (V, E)$ is a DAG, where V are its vertices (nodes) and E are its edges (arrows). T are all its Transition Probability Matrices (TPMs) $T = TPMs(G, D)$.

```

 $E \leftarrow \emptyset$ 
 $T \leftarrow \emptyset$ 
 $B \leftarrow (V, E, T)$ 
 $maxscore \leftarrow -\infty$ 
// DE= all possible directed edges
 $DE = \{\underline{x} \rightarrow \underline{y} \in V \times V : \underline{x} \neq \underline{y}\}$ 
 $again \leftarrow True$ 
while  $again$  do
    for all  $\underline{x} \rightarrow \underline{y} \in DE$  do
        // add arrow
         $E_+ \leftarrow E \cup \{\underline{x} \rightarrow \underline{y}\}$ 
        // delete arrow
         $E_- \leftarrow E - \{\underline{x} \rightarrow \underline{y}\}$ 
        // reverse arrow
         $E_R \leftarrow E_- \cup \{\underline{y} \rightarrow \underline{x}\}$ 
        for  $E' = E_+, E_-, E_R$  do
            if  $E' \neq E$  and  $G' = (V, E')$  is a legal DAG then
                 $T' \leftarrow TPMs(G', D)$ 
                 $B' \leftarrow (G', T')$ 
                 $newscore = \text{BIC-score}(B')$ 
                if  $newscore > maxscore$  then
                     $B \leftarrow B'$ 
                     $maxscore \leftarrow newscore$ 
                else
                     $again \leftarrow False$ 
    return  $B$ 

```

Algorithm 3: Pseudo-code for PC-Stable algorithm

Input : Data D , Vertices (nodes) V , tolerance in CMI $\epsilon > 0$

Output: partially oriented acyclic graph $G = (V, E, UE)$, where V are the vertices (nodes), E are the oriented edges (arrows) and UE are the unoriented edges.

```

 $E \leftarrow \emptyset$ 
// initialize UE to fully-connected undirected graph
 $UE \leftarrow \{\underline{x} - \underline{y} \in V \times V : \underline{x} - \underline{y} = \underline{y} - \underline{x}, \underline{x} \neq \underline{y}\}$ 
// Shrink phase. Deletes edges from E.
 $\text{for } \lambda = 0, 1, 2, \dots, |V| - 2 \text{ do}$ 
     $\quad \text{for all } \underline{x} - \underline{y} \in UE \text{ do}$ 
         $\quad \quad \text{for all } S = \{a \in V : \underline{x} - a \in UE, a \neq \underline{x}, \underline{y}\} \ni |S| = \lambda \text{ do}$ 
             $\quad \quad \quad \text{if } H(\underline{x} : \underline{y} | S) < \epsilon \text{ then}$ 
                 $\quad \quad \quad \quad /* \text{ If there were an arrow between } \underline{x} \text{ and } \underline{y}, \text{ then}$ 
                 $\quad \quad \quad \quad \text{conditioning on } S \text{ would not be enough to interrupt}$ 
                 $\quad \quad \quad \quad \text{info transmission } H(\underline{x} : \underline{y} | S) \text{ between } \underline{x} \text{ and } \underline{y} */$ 
                 $\quad \quad \quad \quad UE \leftarrow UE - \{\underline{x} - \underline{y}\}$ 
                 $\quad \quad \quad \quad S(\underline{x} - \underline{y}) \leftarrow S$ 
     $\quad // Growth phase. Adds v structures to E.$ 
     $\quad \text{for all } \underline{x}, \underline{y}, \underline{a} \text{ such that } \underline{x} - \underline{a} \in UE, \underline{a} - \underline{y} \in UE, \underline{x} - \underline{y} \notin UE, \underline{a} \notin S(\underline{x} - \underline{y}) \text{ do}$ 
         $\quad \quad /* \text{ If there were no collider at } \underline{a}, \text{ then there would be info}$ 
         $\quad \quad \text{transmission between } \underline{x} \text{ and } \underline{y} */$ 
         $\quad \quad UE \leftarrow UE - \{\underline{x} - \underline{a}, \underline{a} - \underline{y}\}$ 
         $\quad \quad E \leftarrow E \cup \{\underline{x} \rightarrow \underline{a}, \underline{y} \rightarrow \underline{a}\}$ 
// Orienting edges.
 $again \leftarrow True$ 
 $size \leftarrow |UE|$ 
 $\text{while } again \text{ do}$ 
     $\quad \text{for all } \underline{x} - \underline{y} \in UE \text{ do}$ 
         $\quad \quad \text{if } \underline{x} \rightarrow \underline{y} \in E, \underline{y} - \underline{z} \in UE, \underline{x} - \underline{z} \notin UE, \nexists \underline{w} \ni \underline{w} \rightarrow \underline{y} \in E \text{ then}$ 
             $\quad \quad \quad // \text{ to avoid introducing new v structure}$ 
             $\quad \quad \quad UE \leftarrow UE - \{\underline{y} - \underline{z}\}$ 
             $\quad \quad \quad E \leftarrow E \cup \{\underline{y} \rightarrow \underline{z}\}$ 
         $\quad \quad \text{if } \underline{x} \rightarrow \underline{y} \in E \text{ and there is directed path from } \underline{x} \text{ to } \underline{y} \text{ in } E \text{ then}$ 
             $\quad \quad \quad // \text{ to avoid introducing cycles}$ 
             $\quad \quad \quad UE \leftarrow UE - \{\underline{x} - \underline{y}\}$ 
             $\quad \quad \quad E \leftarrow E \cup \{\underline{x} \rightarrow \underline{y}\}$ 
     $\quad newsize \leftarrow |UE|$ 
     $\quad \text{if } size == newsize \text{ then}$ 
         $\quad \quad again \leftarrow False$ 
     $\quad \text{else}$ 
         $\quad \quad size \leftarrow newsize$ 


---



496



return  $G = (V, E, UE)$


```

Chapter 71

Support Vector Machines And Kernel Method

This chapter is based on Refs.[102]. [128] and [103].

The Support Vector Machines (SVM) method was first invented with a linear kernel, but was later generalized to arbitrary kernels. We will use the terms SVM method and Kernel Method indistinguishably.

The SVM method is a fairly general method for calculating, via supervised learning, a *binary* classifier. The SVM method finds a continuous surface that separates a space into two disjoint parts.

Let $\Sigma = [0, 1, 2, \dots, nsam - 1]$ be a list of individuals (samples) in a population. In this chapter, we will use the notation $A^\sigma = A[\sigma]$ and $\vec{A} = [A^\sigma : \sigma \in \Sigma]$ for a list (vector, 1-D array) indexed by Σ . We will refer to $DS = (\vec{x}, \vec{y})$ where $x^\sigma \in S_{\underline{x}}$, $y^\sigma \in \{-1, 1\}$, as a dataset. Let $x^\sigma = (x_0^\sigma, x_1^\sigma, \dots, x_{nf-1}^\sigma) \in S_{\underline{x}_0} \times S_{\underline{x}_1} \times \dots \times S_{\underline{x}_{nf-1}} = S_{\underline{x}}$. When $x_j^\sigma \in \mathbb{R}$ for all j , we will take $x^\sigma \in \mathbb{R}^{nf}$ to be a column vector. x^σ is the feature vector for individual σ , and its components x_i^σ for $i = 0, 1, \dots, nf - 1$ are the features. $y^\sigma \in \{-1, 1\}$ is the binary class to which x^σ belongs.

Let $\hat{y}(x^{\sigma_0}) \in \{-1, 1\}$ be an estimate of $y^{\sigma_0} \in \{-1, 1\}$. The **SVM classifier** is defined as

$$\hat{y}(x^{\sigma_0}) = \text{sign}(Y(x^{\sigma_0})) \quad (71.1)$$

where¹

$$Y(x^{\sigma_0}) = \sum_{\sigma} \alpha^\sigma y^\sigma K(x^\sigma, x^{\sigma_0}) . \quad (71.2)$$

The **binary weight coefficients** $\alpha^\sigma \in \{0, 1\}$ for all $\sigma \in \Sigma$ are found by training, via an algorithm to be described below.

The function $K : S_{\underline{x}} \times S_{\underline{x}} \rightarrow \mathbb{R}$ is called the **Kernel or Similarity function**. We assume that $K(x^\sigma, x^{\sigma_0})$ grows bigger when its two arguments x^σ and x^{σ_0} become

¹Define $\text{sign}(0) = 1$.

more “similar”. We also assume that $K(x^\sigma, x^{\sigma_0})$ is symmetric in its two arguments.

71.1 Learning Algorithm for SVM Classifier

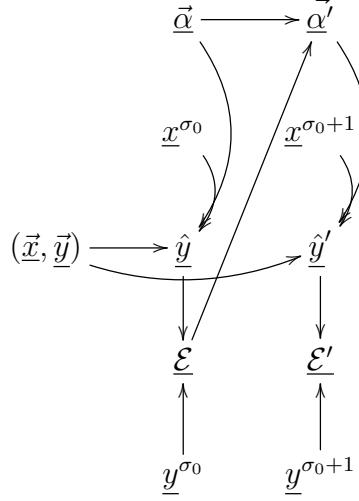


Figure 71.1: Time slice σ_0 of dynamical bnet for learning binary weights $\vec{\alpha}$ of SVM classifier.

Given a kernel function K and a dataset (\vec{x}, \vec{y}) , the SVM classifier is fully specified except for its binary weights $\vec{\alpha}$. Those weights can be learned via the algorithm represented as a causal diagram in Fig.71.1. That figure shows one time slice of a dynamical bnet. The TPMs, printed in blue, of bnet Fig.71.1, are as follows:

$$P(\hat{y}|\vec{\alpha}, (\vec{x}, \vec{y}), x^{\sigma_0}) = \mathbb{1}(\hat{y} = \text{ given by Eq.(71.1).}) \quad (71.3)$$

$$P(\mathcal{E}|\hat{y}, y^{\sigma_0}) = \mathbb{1}(\mathcal{E} = \mathbb{1}(\hat{y} \neq y^{\sigma_0})) \quad (71.4)$$

The first (but not the second, third , etc.) $\vec{\alpha}$ node of Fig.71.1 is a root node. The TPM for that root node should set all components of $\vec{\alpha}$ to zero:

$$P(\vec{\alpha}) = \prod_{\sigma} \mathbb{1}(\alpha^\sigma = 0). \quad (71.5)$$

After that initialization,

$$P(\vec{\alpha}'|\vec{\alpha}, \mathcal{E}) = \mathbb{1}((\alpha')^{\sigma_0} = \alpha^{\sigma_0} + \mathcal{E}) \prod_{\sigma \neq \sigma_0} \mathbb{1}((\alpha')^\sigma = \alpha^\sigma) \quad (71.6)$$

Why this learning algorithm works.

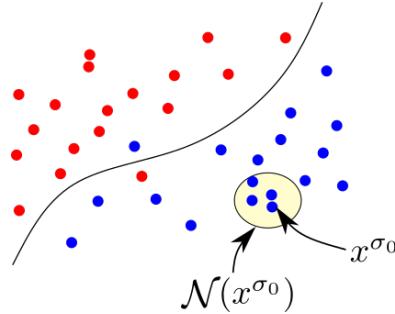


Figure 71.2: Define the neighborhood of x^{σ_0} by $\mathcal{N}(x^{\sigma_0}) = \{x^\sigma : |K(x^\sigma, x^{\sigma_0})| < \epsilon\}$ for some $\epsilon > 0$.

$K(x^\sigma, x^{\sigma_0})$ sets to zero any contribution to $Y(x^{\sigma_0})$ from points x^σ outside the neighborhood $\mathcal{N}(x^{\sigma_0})$ of x^{σ_0} . (See Fig. 71.2). If $\hat{y}(x^{\sigma_0}) = y^{\sigma_0}$, keep $\alpha^{\sigma_0} = 0$ because the neighbors of x^{σ_0} are giving the correct $\hat{y}(x^{\sigma_0})$ when they are polled and the majority wins. If, on the other hand, $\hat{y}(x^{\sigma_0}) \neq y^{\sigma_0}$, then switch α^{σ_0} from 0 to 1, which means x^{σ_0} gets to vote by adding y^{σ_0} to $Y(x^{\sigma_0})$. So we start off with all $\alpha^\sigma = 0$ and we end with most of them still zero except for a select few. If we were to set all α^σ equal to one, we would get overfitting and a very jagged separation between the two classes. The fact that we end with only a select few α^σ equal to 1, and the rest equal to 0, helps make the demarcation between the two classes less jagged.

71.2 Linear (dot-product) Kernel

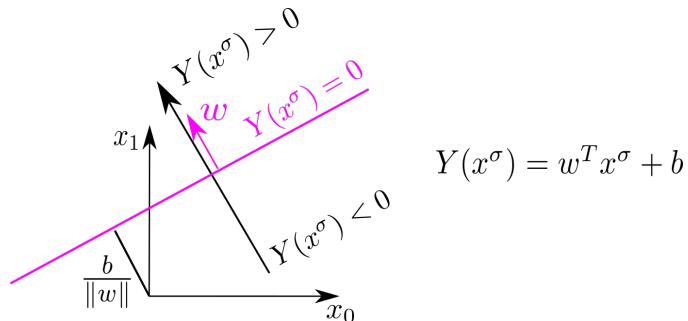


Figure 71.3: Graph of line $Y(x^\sigma) = 0$ splits plane into regions with $Y < 0$, $Y = 0$ and $Y > 0$.

So far, we have discussed the SVM method for an arbitrary kernel. This section is devoted to the **Linear (aka dot-product) Kernel**. Said kernel is defined as

$$K(x^\sigma, x^{\sigma_0}) = (x^\sigma)^T x^{\sigma_0} . \quad (71.7)$$

For this kernel, Eq.(71.2) specializes to

$$Y(x^{\sigma_0}) = \sum_{\sigma} \alpha^{\sigma} y^{\sigma} K(x^{\sigma}, x^{\sigma_0}) + b \quad (71.8)$$

$$= w^T x^{\sigma_0} + b \quad (71.9)$$

where

$$w = \sum_{\sigma} \alpha^{\sigma} y^{\sigma} x^{\sigma}. \quad (71.10)$$

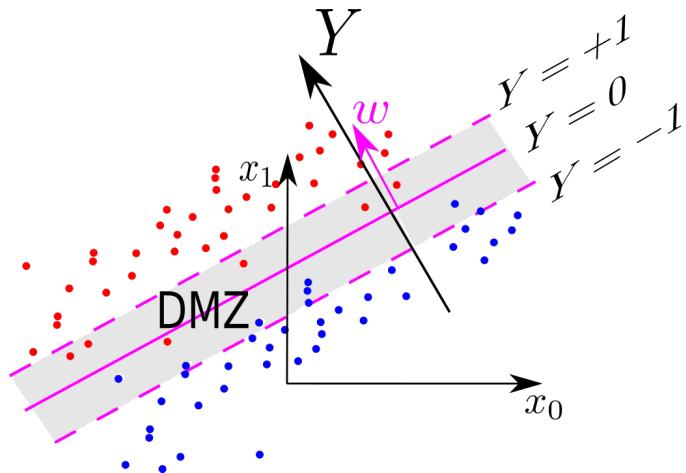


Figure 71.4: We refer to the gray shaded region with $-1 < Y < 1$, where $Y = w^T x + b$, as the DMZ.

We started this chapter by pulling the SVM classifier out of a hat. We did give reasons why it works, but we did not derive it from a more general minimization principle. Such a derivation is possible, at least in the linear kernel case, and we give it next.

Consider the following 3 straight lines:

$$w^T x^{\sigma} + b = +A \quad (71.11)$$

$$w^T x^{\sigma} + b = 0 \quad (71.12)$$

$$w^T x^{\sigma} + b = -A \quad (71.13)$$

where $w, x^{\sigma} \in \mathbb{R}^{nf}$, and $b, A \in \mathbb{R}$. We can re-scale the vector w and scalar b so as to get rid of the A . (i.e., replace $w \rightarrow wA$ and $b \rightarrow bA$ and divide common factor A out of equations). This rescaling does not affect the graphs (i.e., x loci) of these 3 lines. Now we have:

$$w^T x^\sigma + b = +1 \quad (71.14)$$

$$w^T x^\sigma + b = 0 \quad (71.15)$$

$$w^T x^\sigma + b = -1 \quad (71.16)$$

If Y stands for

$$Y = w^T x^\sigma + b, \quad (71.17)$$

then we define the **DMZ (demilitarized zone)** to be the region

$$DMZ = \{x^\sigma : |Y(x^\sigma)| < 1\}. \quad (71.18)$$

The lines $Y = \pm 1$ will be called the **borders (aka margins)** of the DMZ, and line $Y = 0$ will be called the **line of demarcation** of the DMZ. The DMZ is illustrated in Fig.71.4.

Let D_{DMZ} be the **DMZ width** (i.e., the distance from one border of the DMZ to the other.) Position vectors pointing from the origin to either of the two DMZ borders are called **support vectors**. Suppose $X^+, X^- \in \mathbb{R}^{nf}$ are two support vectors on opposite DMZ borders with $|X^+ - X^-| = D_{DMZ}$. Then

$$w^T X^+ + b = 1 \quad (71.19)$$

$$w^T X^- + b = -1 \quad (71.20)$$

so

$$D_{DMZ} = \frac{2}{|w|}. \quad (71.21)$$

For any $a \in \mathbb{R}$, let the **positive a_+ and negative a_- parts of a** be given by

$$a = \underbrace{a_+}_{a \mathbb{1}(a>0)} + \underbrace{a_-}_{a \mathbb{1}(a \leq 0)}. \quad (71.22)$$

When $Y = \pm 1$, an error in $Y(x^\sigma)$ occurs iff $y^\sigma Y(x^\sigma) = -1$. But how should we define errors when Y is a real number? Define the **Cost of erring for sample σ to be**

$$CE^\sigma(x^\sigma, y^\sigma) = (1 - y^\sigma Y(x^\sigma))_+. \quad (71.23)$$

CE^σ is shown in Fig.71.5. As you can see, there is a penalty for living on the incorrect side, and even a penalty for living on the correct side but too close to the DMZ.

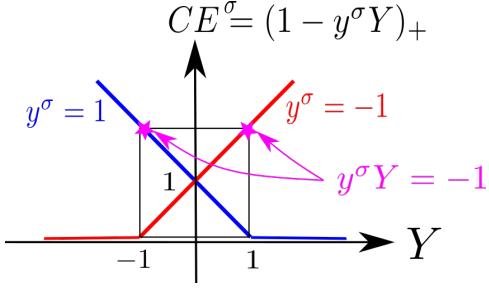


Figure 71.5: Plot of CE^σ versus Y .

Note that the line of demarcation should have the lowest CE^σ for all possible w, b . So to find that line, we want to minimize CE^σ with respect to w, b . But note that $CE^\sigma \geq 0$, and it can be zero for an appropriately chosen w . So we need to add another cost in order to get a non-zero total cost. Define the **DMZ cost** as

$$CZ = \frac{1}{2}|w|^2 = \frac{2}{D_{DMZ}^2} \quad (71.24)$$

Note that $CZ \rightarrow \infty$ as $D_{DMZ} \rightarrow 0$, so CZ penalizes DMZ's that are too narrow.

Now define a Lagrangian \mathcal{L} to be the sum of these 2 contributions.

$$\mathcal{L} = CZ + \sum_{\sigma} CE^\sigma \quad (71.25)$$

$$= \frac{1}{2}|w|^2 + \sum_{\sigma} (1 - y^{\sigma}Y(x^{\sigma}))_{+} \quad (71.26)$$

This particular choice of CZ is not unique, but it isn't totally arbitrary either. We want it to be independent of the sample σ , and to depend on a geometrical aspect of the DMZ, like its width $D_{DMZ} = 2/|w|$. Note that CE^σ behaves, when $|w| \rightarrow \infty$, linearly in $|w|$. We are going to differentiate \mathcal{L} with respect to $|w|$ to find an optimum. But straight lines have no optima, so we need CZ to behave, when $|w| \rightarrow \infty$, as $|w|^p$ for some integer $p > 1$.

Setting the variation $\delta\mathcal{L}$ to zero, we get

$$0 = \delta\mathcal{L} = \delta w_i \left\{ w_i - \sum_{\sigma} \mathbb{1}(y^{\sigma}Y(x^{\sigma}) < 1) y^{\sigma} x_i^{\sigma} \right\} \quad (71.27)$$

so

$$w = \sum_{\sigma} \underbrace{\mathbb{1}(y^{\sigma}Y(x^{\sigma}) < 1)}_{\alpha^{\sigma}} y^{\sigma} x^{\sigma}. \quad (71.28)$$

71.3 Alternatives to Linear Kernel

Sometimes it is convenient to replace the dot-product kernel given above by a different kernel. Other popular kernels are:

- **Radial Basis Function (RBF) Kernel.** In this case, K is a radial function; i.e., a function that depends only on the magnitude (radius, Euclidean distance, L^2 norm) of a vector. An example of an RBF kernel is the **Gaussian Kernel**

$$K(x^\sigma, x^{\sigma_0}) = e^{-\gamma|x^\sigma - x^{\sigma_0}|^2} \quad (71.29)$$

for some free parameter $\gamma > 0$.

- **Inhomogeneous Polynomial Kernel**

$$K(x^\sigma, x^{\sigma_0}) = [(x^\sigma)^T x^{\sigma_0} + 1]^d \quad (71.30)$$

for some positive integer d .

- **Homogeneous Polynomial Kernel**

$$K(x^\sigma, x^{\sigma_0}) = [(x^\sigma)^T x^{\sigma_0}]^d \quad (71.31)$$

for some positive integer d .

- **"Kernel trick" Kernel.** Consider a map $\Phi : \mathbb{R}^{nf} \rightarrow \mathbb{R}^N$. Usually $N > nf$. Φ can be a rectangular matrix $A \in \mathbb{R}^{N \times nf}$ so that $\Phi(x^\sigma) = Ax^\sigma \in \mathbb{R}^N$. Let

$$K(x^\sigma, x^{\sigma_0}) = [\Phi(x^\sigma)]^T \Phi(x^{\sigma_0}) \quad (71.32)$$

Although the constant surfaces of this kernel are hyperplanes in \mathbb{R}^N , its constant surfaces in \mathbb{R}^{nf} can be curved and even closed compact surfaces (e.g. spheres).

71.4 Random Forest and Kernel Method

Chapter 72

Synthetic Controls

This chapter is based on Refs.[8] and [6].

This chapter assumes that the reader has read Chapter 15 on the Difference-in-Differences (DID) method.

The Synthetic Controls (SC) method is a simple enhancement of the DID method. SC enhances DID in two simple yet powerful ways:

1. **Better time resolution.** DID considers just 2 time-snapshots (i.e., a time-series with only 2 times) whereas SC considers arbitrarily many time-snapshots (i.e., a time-series with more than 2 times).
2. **Weighted average of controls.** DID divides the population of individuals into just 2 kinds: the treated and the untreated (aka controls). SC divides the total population into treated and controls just like DID does, but it goes further and divides the control population into multiple subpopulations, and calculates a weighted average, called a “synthetic control”, of those subpopulations. The weights of the synthetic control are chosen so that it mimics as closely as possible the behavior of the treated population for all times measured before the treatment was applied.

Let us describe these two enhancements more precisely.

- **timing:** Let t_k for $k = 0, 1, \dots, n_{pre} - 1$ be the pre-treatment times at which a measurement occurs. Let t_k for $k = n_{pre}, n_{pre} + 1, \dots, n_t - 1$ be the post-treatment times at which a measurement occurs. Note that $n_{pre} + n_{post} = n_t$. Note that $t_* = t_{n_{pre}+1}$ is the first measurement time after the treatment is applied, t_0 is the first measurement time, and $t_{fin} = t_{n_t-1}$ is the last one.
- **subpopulations:** Let $S_1 = \{\sigma_1\}$ be the set of treated units (just one). Let $S_0 = \{\sigma : \sigma \neq \sigma_1\}$ be the set of untreated units (i.e., controls). Let $nsam$ = number of all units σ , $n_1 = |S_1| = 1$, and $n_0 = |S_0| = nsam - 1$.
- **weights:**

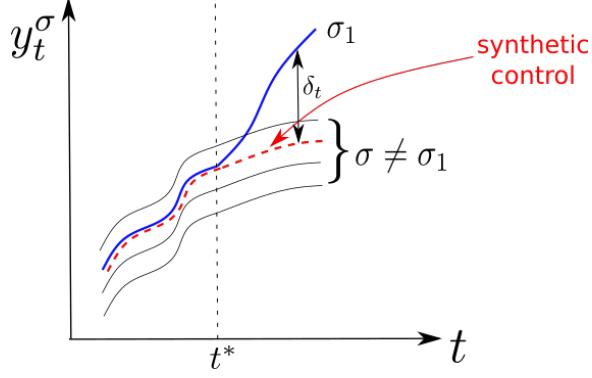


Figure 72.1: Pictorial representation of the Synthetic Controls (SC) method. The outcome y of the synthetic control unit is colored red and that of the treated unit is colored blue. They roughly agree for $t < t_*$.

We want to define a time-independent weight w^σ for each unit σ in such a way that the output y_t^σ for the synthetic control unit behaves like the output for the treated unit σ_1 for $t < t_*$.

Let

$$w^{\sigma_1} = 0 \quad (72.1)$$

and

$$w^{n_0} = \{w^\sigma\}_{\sigma \neq \sigma_1}. \quad (72.2)$$

Define a cost function \mathcal{C} :

$$\mathcal{C}(w^{n_0}) = \sum_{t < t_*} \left(y_t^{\sigma_1} - \sum_{\sigma \neq \sigma_1} w^\sigma y_t^\sigma \right)^2 \quad (72.3)$$

Then calculate w^{n_0} by minimizing the cost function, subject to the constraint that w^{n_0} be a probability distribution:

$$w^{n_0} = \underset{W^{n_0}}{\operatorname{argmin}} \left\{ \mathcal{C}(W^{n_0}) : W^\sigma \geq 0, \sum_{\sigma \neq \sigma_1} W^\sigma = 1 \right\}. \quad (72.4)$$

Now that we have defined a weight w^σ for every unit σ , we can define for $c \in \{0, 1\}$,

$$y_t^{\sigma_1}(c) = \begin{cases} y_t^{\sigma_1} & \text{if } c = 1 \\ \sum_{\sigma \neq \sigma_1} w^\sigma y_t^\sigma & \text{if } c = 0 \end{cases} \quad (72.5)$$

$$\mathcal{Y}_c(t) = E_\sigma[y_t^\sigma(c)] \quad (72.6)$$

and

$$\delta_t = \mathcal{Y}_1(t) - \mathcal{Y}_0(t) \quad (72.7)$$

δ_t is illustrated in Fig.72.1. It approximates $ATE(t)$.

72.1 PO analysis

In this section, we show how to analyze the SC method using the formalism of PO theory.

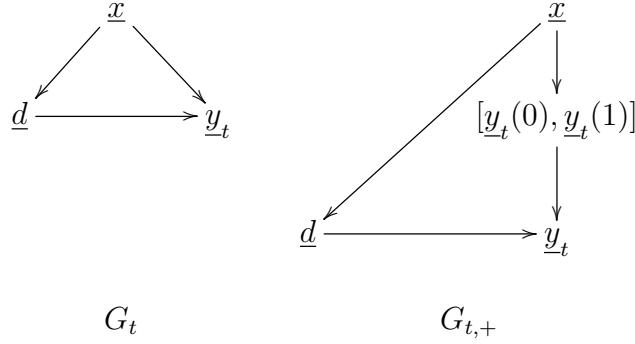


Figure 72.2: $t \in \{t_0, t_1, \dots, t_{fin}\}$. Bnet $G_{t,+}$ is obtained by adding two new nodes $y_t(0)$ and $y_t(1)$ to bnet G_t .

As usual for PO theory, we will consider expected values of y_t^σ :

$$E_{\sigma|d,x}[y_t^\sigma(c)] = E_{y_t(c)|d,x}[y_t(c)] = \mathcal{Y}_{c|d,x}(t) \quad (72.8)$$

To calculate these expected values, we need a “model” with probability distributions. In this case, the needed model and probability distributions are provided by the bnets depicted in Fig.72.2. The TPMs, printed in blue, for the bnet $G_{t,+}$ in Fig.72.2, are as follows. Note that the TPMs for the bnet $G_{t,+}$ are defined in terms of the TPMs for the bnet G_t .

$$P(x) = P_{\underline{x}}(x) \quad (72.9)$$

$$P(d|x) = P_{d|\underline{x}}(d|x) \quad (72.10)$$

$$P(y_t|y_t(0), y_t(1), d) = \mathbb{1}(y_t = y_t(d)) \quad (72.11)$$

$$P(y_t(c)|x) = P(y_t(c)|d, x) = \text{given} \quad (72.12)$$

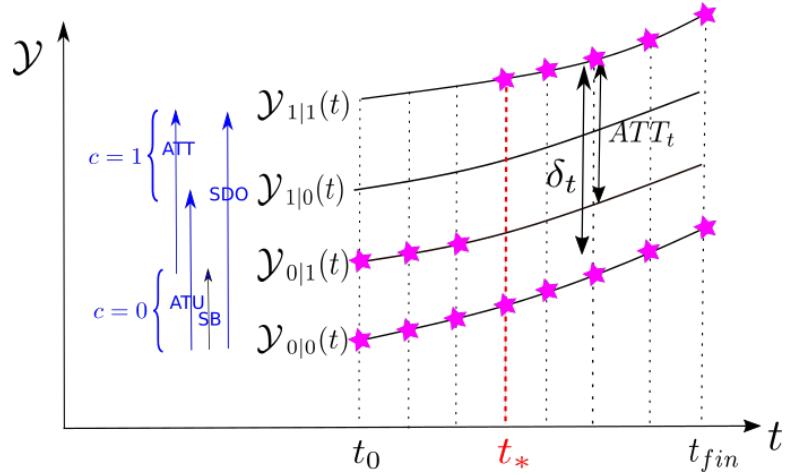


Figure 72.3: Four different time-dependent expected values $\mathcal{Y}_{c|d}(t)$ of y_t^σ for bnet $G_{t,+}$. The $2 * nt$ magenta stars represents the $2 * nt$ SC measurements.

Fig.72.3 depicts the four functions $\mathcal{Y}_{c|d}(t)$ for t in the interval $[t_0, t_{fin}]$ and for $c, d \in \{0, 1\}$. The \mathcal{Y} coordinates of the $2 * nt$ magenta stars in Fig.72.3 can be calculated using bnet G_t . Note that in Fig.72.3, we display a large gap between the curves $\mathcal{Y}_{0|d}(t)$ for $d \in \{0, 1\}$. In reality, $P(y_t(0)|d)$ has been constructed so as to make that gap as small as possible. Thus, to a good(?) approximation,

$$\delta_t \approx ATE_t \quad (72.13)$$

Unlike in the DID method, in the SC method, to a good(?) approximation, we don't have to worry about parallel trends.

Chapter 73

Time Series Analysis: ARMA and VAR

This chapter is based mostly on the book Ref.[14] on time series analysis by Hamilton, and on the lectures Ref. [19] by Chung-Ming Kuan. In writing this chapter, we also profited greatly from numerous Wikipedia entries on time series analysis, such as the entries on time series (Ref.[129]), ARMA time series (Ref.[67]), AR time series (Ref.[66]), MA time series (Ref.[112]), and VAR time series (Ref.[134]).

We cover only a small fraction of the treasures covered in those sources, and only cover stationary time-series. Non-stationary time series we don't even touch. But we hope to have covered enough to pique our readers's interest in time series analysis, and make him/her appreciate how bnets make time series much more intuitive and fun. The time-series considered in this chapter can be represented by one of the simplest types of bnets, namely, the LDEN bnets introduced in Chapter 38.

As usual, for $t, t_a, t_b \in \mathbb{Z}$, let $\mathbb{Z}_{<t} = \{t-1, t-2, t-3, \dots\}$, $\mathbb{Z}_{[t_a, t_b]} = \{t_a, t_a+1, \dots, t_b\}$, etc.

Let $\underline{x}_t \in \mathbb{R}$. A **time series (t-series)**, denoted variously by $\{\underline{x}_1, \underline{x}_2, \dots, \underline{x}_{nt}\} = \{\underline{x}_t\}_{t=1}^{nt} = \{\underline{x}_t\}_{t \in \mathbb{Z}}$, is a set of real numbers index by a discrete set of times $\mathbb{Z}_{[0, nt]}$.

For $t_a < t_b$, let $\underline{x}_{[t_a, t_b]} = (\underline{x}_{t_a}, \underline{x}_{t_a+1}, \dots, \underline{x}_{t_b})$. Let $x_{<t} = (\dots, x_{t-2}, \underline{x}_{t-1})$.

73.1 White noise

By **white noise** $\{\underline{n}_t\}_{\forall t} \sim WN(0, \sigma^2)$ we mean a t-series $\{\underline{n}_t\}_{\forall t}$ that satisfies

$$E[\underline{n}_t] = 0 \tag{73.1}$$

and

$$\langle \underline{n}_t, \underline{n}_{t'} \rangle = \sigma^2 \delta(t, t') . \tag{73.2}$$

Gaussian white noise $\{\underline{n}_t\}_{\forall t} \sim WN(0, \sigma^2)_{\mathcal{N}}$ is white noise such that also $\underline{n}_t \sim \mathcal{N}(0, \sigma^2)$.

73.2 Backshift operator

\mathcal{B} is the backshift (a.k.a. lag) operator. For any t-series $\{\underline{x}_t\}_{\forall t}$, $\{\underline{y}_t\}_{\forall t}$ and scalars $a, b \in \mathbb{R}$,

$$\mathcal{B}\underline{x}_t = \underline{x}_{t-1} \quad (73.3)$$

$$\mathcal{B}(ax_t + by_t) = a\mathcal{B}(x_t) + b\mathcal{B}(y_t) \quad (73.4)$$

A \mathcal{B} Polynomial with coefficients $\alpha_{[0,p]}$:

$$\alpha(\mathcal{B}) = \alpha_0 + \alpha_1\mathcal{B} + \alpha_2\mathcal{B}^2 + \dots + \alpha_p\mathcal{B}^p \quad (73.5)$$

\mathcal{B}^{-1} is the inverse of the backshift operator (a.k.a. frontshift operator)

$$\mathcal{B}^{-1}\underline{x}_t = \underline{x}_{t+1} \quad (73.6)$$

The following two Taylor expansions prove useful in finding the inverse of backshift operator polynomials:

- $$\frac{1}{1-z} = 1 + z + z^2 + \dots \quad (73.7)$$

converges for $z \in \mathbb{C}$ with $|z| < 1$. We will use this expansion with z replaced by $\alpha\mathcal{B}$, where $\alpha \in \mathbb{R}$.

- $$\frac{1}{1-z} = (-z^{-1}) \left[\frac{1}{1-z^{-1}} \right] = (-z^{-1}) [1 + z^{-1} + z^{-2} + \dots] \quad (73.8)$$

converges for $z \in \mathbb{C}$ with $|z| > 1$. We will use this expansion with z^{-1} replaced by $(\alpha\mathcal{B})^{-1}$, where $\alpha \in \mathbb{R}$.

73.3 Metrics

Consider a t-series $\{\underline{x}_t\}_{\forall t}$.

In general, if we have a metric like Auto-covariance (ACov) that is defined for $\tau = 1, 2, 3, \dots$, it is conventional in time series analysis to refer to the plot of that metric for all values of τ as the Auto-covariance *Function* (ACovF).

- Expected value and Variance

$$E[\underline{x}_t] \quad (73.9)$$

$$Var[\underline{x}_t] = \langle \underline{x}_t, \underline{x}_t \rangle \quad (73.10)$$

- **Auto-covariance (ACov)**

$$\gamma_{t,t+\tau} = \langle \underline{x}_t, \underline{x}_{t+\tau} \rangle \quad (73.11)$$

- **Auto-correlation (ACorr)** (assumes w-stationarity)

$$\rho(\tau) = \frac{\gamma(\tau)}{\gamma(0)} \quad (73.12)$$

- **Generating function of auto-covariance** (assumes w-stationarity)

$$\tilde{\gamma}(z) = \sum_{\tau=-\infty}^{\infty} \gamma(\tau) z^{\tau} \quad (73.13)$$

Note that this transform is double sided. Fourier Transform if $z = e^{-i\omega\tau}$.

- **Expected value and variance conditioned on all past information**

For $\tau = 1, 2, 3, \dots$,

$$E_{|x_{\leq t}}[\underline{x}_{t+\tau}] \quad (73.14)$$

$$Var_{|x_{\leq t}}[\underline{x}_{t+\tau}] = \langle \underline{x}_{t+\tau}, \underline{x}_{t+\tau} \rangle_{|x_{\leq t}} \quad (73.15)$$

- **Partial auto-covariance (PACov)**

Assume w-stationarity. For $\tau = 1, 2, 3, \dots$

$$\gamma^{part}(\tau) = \langle \underline{x}_t, \underline{x}_{t+\tau} \rangle_{|x_{\leq t}, \underline{x}_{t+\tau}} \quad (73.16)$$

The idea is that we set to zero the nodes $\underline{x}_{(t,t+\tau)} = \{\underline{x}_{t+1}, \underline{x}_{t+2}, \dots, \underline{x}_{t+\tau-1}\}$ that lie between (but not including) \underline{x}_t and $\underline{x}_{t+\tau}$.

- **Partial auto-correlation (PACorr)**

$$\rho^{part}(\tau) = \frac{\gamma^{part}(\tau)}{\gamma^{part}(0)} \quad (73.17)$$

weak stationarity (w-stationarity) means that $E[\underline{x}_t] = \mu$ and $\gamma_{t,t+\tau} = \gamma(\tau)$ are both independent of t . If we have w-stationarity, then

$$\gamma(-\tau) = \langle \underline{x}_t, \underline{x}_{t-\tau} \rangle \quad (73.18)$$

$$= \langle \underline{x}_{t-\tau}, \underline{x}_t \rangle \quad (73.19)$$

$$= \gamma(\tau) \quad (73.20)$$

We will often abbreviate $\gamma(\tau)$ by γ_τ .

Example of various metrics. If $\{\underline{n}_t\}_{\forall t} \sim WN(0, \sigma^2)$ then

$$E[\underline{n}_t] = 0 \quad (73.21a)$$

$$\gamma(\tau) = \sigma^2 \delta(\tau, 0) \quad (73.21b)$$

$$\gamma(0) = \sigma^2 \quad (73.21c)$$

$$\tilde{\gamma}(z) = \gamma(0) \quad (73.21d)$$

For $\tau > 0$,

$$E_{|\underline{n}_{\leq t}}[\underline{n}_{t+\tau}] = E[\underline{n}_{t+\tau}] = 0 \quad (73.21e)$$

$$\langle \underline{n}_{t+\tau}, \underline{n}_{t+\tau} \rangle_{|\underline{n}_{\leq t}} = E[\underline{n}_{t+\tau}^2] = \sigma^2 \quad (73.21f)$$

73.4 Definition of $ARMA(p, q)$, $AR(p)$ and $MA(q)$.

Suppose $\{\underline{y}_t\}_{\forall t}$ is a zero mean t-series. Hence $\underline{y}_t = \underline{Y}_t - \mu$, $E[\underline{Y}_t] = \mu$, $E[\underline{y}_t] = 0$. \underline{y}_t is said to be the **demeaned** version of \underline{Y}_t .

Suppose also that $\{\underline{n}_t\}_{\forall t} \sim WN(0, \sigma^2)$.

Then we define the **Auto-Regressive Moving-Average t-series** $ARMA(p, q)$ by

$$\underline{y}_t = \underbrace{\sum_{j=1}^p \alpha_j \underline{y}_{t-j}}_{\mathcal{Y}_t^{AR(p)}} + \underline{n}_t + \underbrace{\sum_{j=1}^q \nu_j \underline{n}_{t-j}}_{\mathcal{Y}_t^{MA(q)}} \quad (73.22)$$

(α stands for the first letter of “auto-regressive”. ν stands for first letter of “noise”.)

Special cases

1. Auto-Regressive t-series $AR(p)$

$$\underline{y}_t = \mathcal{Y}_t^{AR(p)} + \underline{n}_t \quad (73.23)$$

2. Moving-Average t-series $MA(q)$

$$\underline{y}_t = \underline{n}_t + \mathcal{Y}_t^{MA(q)} \quad (73.24)$$

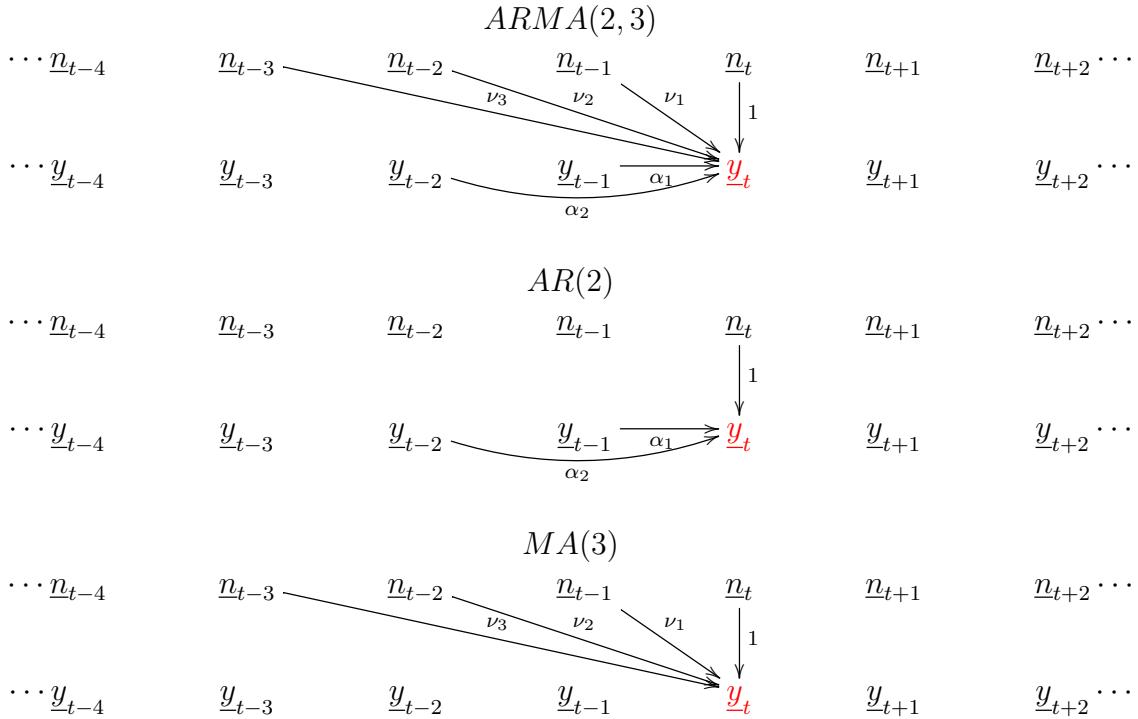


Figure 73.1: $ARMA(2,3)$, $AR(2)$ and $MA(3)$ bnets. For clarity, we show only the arrows entering node y_t . The full bnet has the same structural pattern of incoming arrows (including the weights α_j, ν_j) for each node $y_{t'}$ for all t' .

Fig.73.1 shows the bnets for $ARMA(2,3)$, $AR(2)$ and $MA(3)$. The TPM, printed in blue, for node y_t in those bnets, is as follows:

For $ARMA(p,q)$,

$$P(y_t | y_{[t-p,t-1]}, n_{[t-q,t]}) = \mathbb{1}(y_t = \text{see Eq.73.22})) \quad (73.25)$$

For $AR(p)$,

$$P(y_t | y_{[t-p,t-1]}, n_t) = \mathbb{1}(y_t = \text{see Eq.73.23})) \quad (73.26)$$

For $MA(q)$,

$$P(y_t | n_{[t-q,t]}) = \mathbb{1}(y_t = \text{see Eq.73.24})) \quad (73.27)$$

The n_t variable is variously referred to as the **external noise**, **impulse**, **shock**, **innovation** at time t .

73.5 Solving $AR(p)$

Suppose $\{\underline{y}_t\}_{\forall t}$ is an $AR(p)$ t-series. Hence

$$\underline{y}_t = \sum_{j=1}^p \alpha_j \underline{y}_{t-j} + \underline{n}_t \quad (73.28)$$

$AR(0)$ satisfies:

$$\underline{y}_t = \underline{n}_t \quad (73.29)$$

See Fig.73.2. This is just white noise.

$AR(1)$ satisfies:

$$\underline{y}_t = \alpha_1 \underline{y}_{t-1} + \underline{n}_t \quad (73.30)$$

See Fig.73.2. This is a Markov chain with external i.i.d. noise injected to each node. $AR(1)$ is the discrete form of the so called **Ornstein-Uhlenbeck t-series** (aka as the **Langevin Equation**). When $\alpha_1 = 1$, it is called a **random walk**.

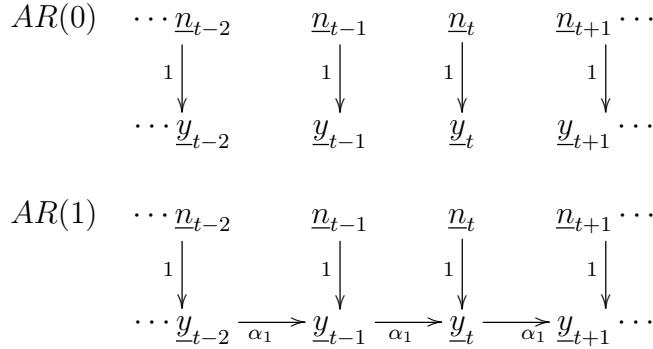


Figure 73.2: BNets for $AR(0)$ and $AR(1)$.

Note that

$$\alpha^-(\mathcal{B}) \underline{y}_t = \underline{n}_t \quad (73.31)$$

where

$$\alpha^-(\mathcal{B}) = 1 - \sum_{j=1}^p \alpha_j \mathcal{B}^j \quad (73.32)$$

Note that we can get $AR(p)$ from $AR(\infty)$ by setting $\alpha_{>p} = 0$.

If $\alpha^-(\beta)$ is invertible, then, using the Taylor expansion Eq.(73.7), we get

$$\underline{y}_t = \alpha'(\mathcal{B}) \underline{n}_t \quad (73.33)$$

where

$$\alpha'(\mathcal{B}) = \frac{1}{\alpha^-(\mathcal{B})} = 1 + \sum_{k=1}^{\infty} \left[\sum_{j=1}^p \alpha_j \mathcal{B}^j \right]^k = \sum_{j=0}^{\infty} \alpha'_j \mathcal{B}^j \quad (73.34)$$

where $\alpha'_0 = 1$.

73.6 Solving $MA(q)$

Suppose $\{\underline{y}_t\}_{\forall t}$ is an $MA(q)$ t-series. Hence,

$$\underline{y}_t = \underline{n}_t + \sum_{j=1}^q \nu_j \underline{n}_{t-j} = \sum_{j=0}^q \nu_j \underline{n}_{t-j} \quad (73.35)$$

where $\nu_0 = 1$. Thus,

$$\underline{y}_t = \nu(\mathcal{B}) \underline{n}_t \quad (73.36)$$

where

$$\nu(\mathcal{B}) = 1 + \sum_{j=1}^q \nu_j \mathcal{B}^j = \sum_{j=0}^q \nu_j \mathcal{B}^j . \quad (73.37)$$

Note that we can get $MA(q)$ from $MA(\infty)$ by setting $\nu_{>q} = 0$.

Claim 97 *If $\{\underline{y}_t\}_{\forall t}$ is an $MA(q)$ t-series, then*

$$E[\underline{y}_t] = 0 \quad (73.38a)$$

For $\tau \geq 0$,

$$\gamma(\tau) = \mathbb{1}(\tau \leq q) \sigma^2 \sum_{j=0}^{q-\tau} \nu_j \nu_{\tau+j} \quad (73.38b)$$

$$\gamma(0) = \sigma^2 \sum_{j=0}^q \nu_j^2 \quad (73.38c)$$

proof:

$$\gamma(\tau) = \left\langle \underline{y}_t, \underline{y}_{t+\tau} \right\rangle \quad (73.39)$$

$$= \left\langle \sum_{j=0}^q \nu_j \underline{n}_{t-j}, \sum_{k=0}^q \nu_k \underline{n}_{t+\tau-k} \right\rangle \quad (73.40)$$

$$= \sigma^2 \sum_{j=0}^q \sum_{k=0}^q \nu_j \nu_k \delta(t-j, t+\tau-k) \quad (73.41)$$

$$= \sigma^2 \sum_{j=0}^q \sum_{k=0}^q \nu_j \nu_k \delta(k, \tau+j) \quad (73.42)$$

$$= \mathbb{1}(\tau \leq q) \sigma^2 \sum_{j=0}^{q-\tau} \nu_j \nu_{\tau+j} \quad (73.43)$$

QED

73.7 Solving $ARMA(p, q)$

Suppose $\{\underline{y}_t\}_{\forall t}$ is an $ARMA(p, q)$ t-series. Hence, using Eqs.(73.31) and (73.36), \underline{y}_t satisfies

$$\alpha^-(\mathcal{B})\underline{y}_t = \nu(\mathcal{B})\underline{n}_t \quad (73.44)$$

If $\alpha^-(\beta)$ is invertible, then, using the Taylor expansion Eq.(73.7), we get

$$\underline{y}_t = \frac{\nu(\mathcal{B})}{\alpha^-(\mathcal{B})}\underline{n}_t = \nu(\mathcal{B})\alpha'(\mathcal{B})\underline{n}_t \quad (73.45)$$

We see that, if the $\alpha^-(\mathcal{B})$ operator is invertible, an $AR(p)$ or an $ARMA(p, q)$ t-series can be represented as an $MA(\infty)$ t-series. $MA(\infty)$ is often called **Wold's Decomposition**. Furthermore, if the $\nu(\mathcal{B})$ operator is invertible, an $MA(q)$ or an $ARMA(p, q)$ t-series can be represented as an $AR(\infty)$ t-series.

The polynomials $\alpha^-(z)$ and $\nu(z)$ can be expressed in factored form $\alpha^-(z) = \prod_{j=1}^p (z - z_j^\alpha)$ and $\nu(z) = \prod_{j=1}^q (z - z_j^\nu)$. If these two polynomials have a root z_0 in common, both polynomials should be divided by $(z - z_0)$. This reduces an $ARMA(p, q)$ t-series to an $ARMA(p-1, q-1)$ t-series. The bnet for $ARMA(p-1, q-1)$ has one less α_j arrow and one less ν_j arrow than the bnet for $AR(p, q)$.

73.8 Auto-correlation and partial auto-correlation

Note from Eq.(73.38b) that if $\{\underline{y}_t\}_{\forall t}$ is a $MA(q)$ t-series, then $\gamma(\tau) = 0$ for all $\tau > q$. Fig.73.3 gives a graphical proof, using bnets and the d-separation theorem, that for

a $MA(2)$ t-series, $\gamma(\tau) = 0$ for $\tau > 2$. As a consequence of this, a plot of $\gamma(\tau)$ versus τ for a typical $MA(2)$ t-series looks like Fig.73.4.

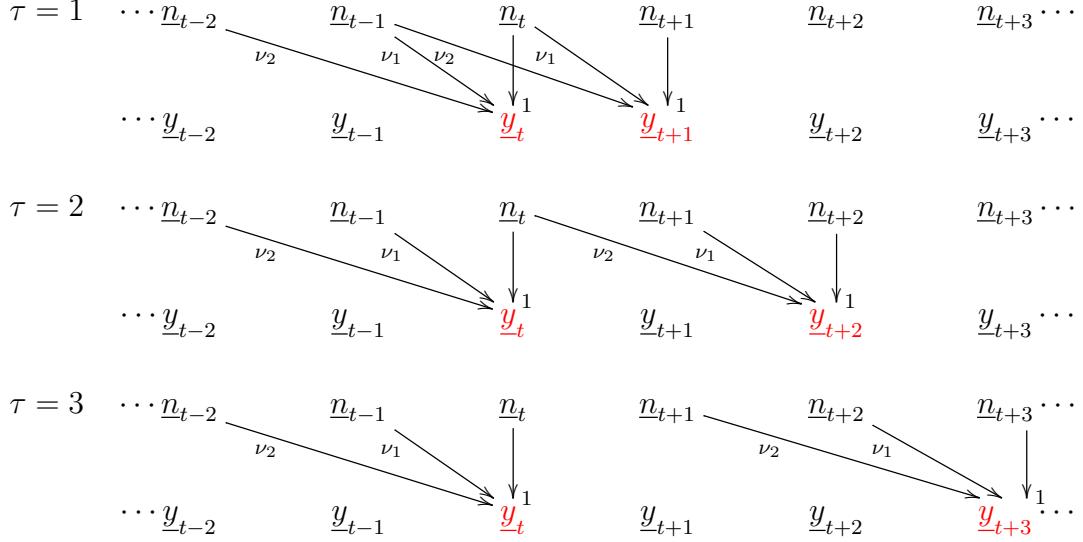


Figure 73.3: $MA(2)$ bnet. For clarity, we show only arrows entering nodes \underline{y}_t and $\underline{y}_{t+\tau}$ for $\tau = 1, 2, 3$. For $\tau = 1, 2$, there is a path through which information can flow from node \underline{y}_t to node $\underline{y}_{t+\tau}$. For $\tau = 2$, there is no such path.

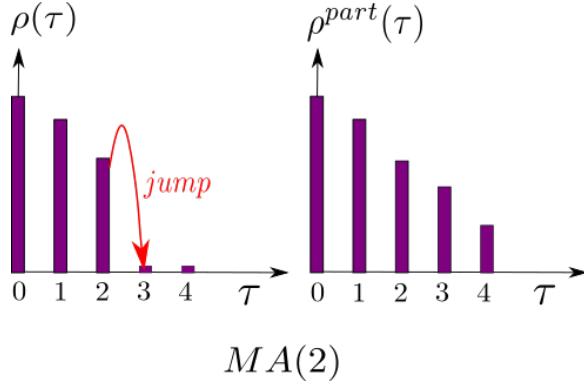


Figure 73.4: Plot of auto-correlation function (ACorrF) $\rho(\tau)$ and partial auto-correlation function (PACorrF) $\rho^{part}(\tau)$ for an instance of $MA(2)$. Note that $\rho(\tau)$ vanishes for $\tau > 2$.

Claim 98 If $\{\underline{y}_t\}_{\forall t}$ is an $AR(p)$ t-series, then $\gamma^{part}(\tau) = 0$ for all $\tau > p$.

proof:

Let

$$\underline{\xi} = \underline{y}_{\leq t}, \underline{y}_{t+\tau} \quad (73.46)$$

Recall that

$$\gamma^{part}(\tau) = \left\langle \underline{y}_t, \underline{y}_{t+\tau} \right\rangle_{|\xi} \quad (73.47)$$

$$= \left\langle \underline{y}_t \underline{y}_{t+\tau} \right\rangle_{|\xi} - \left\langle \underline{y}_t \right\rangle_{|\xi} \left\langle \underline{y}_{t+\tau} \right\rangle_{|\xi} \quad (73.48)$$

$$= \left\langle \underline{y}_t \underline{y}_{t+\tau} \right\rangle_{|\xi} \quad (73.49)$$

Define

$$Z_{(t,t+\tau)} = \mathbb{1}(\underline{y}_{(t,t+\tau)} = 0) \quad (73.50)$$

Hence, the operator $Z_{(t,t+\tau)}$ sets all $\underline{y}_{t'}$ with $t' < t + \tau$ equal to zero. Note that we can express the PACov as

$$\gamma^{part}(\tau) = \left\langle Z_{(t,t+\tau)} (\underline{y}_t \underline{y}_{t+\tau}) \right\rangle. \quad (73.51)$$

Since

$$\underline{y}_{t+\tau} = \sum_{j=1}^p \alpha_j \underline{y}_{t+\tau-j} + \underline{n}_{t+\tau} \quad (73.52)$$

$$= \alpha_p \underline{y}_{t+\tau-p} + \dots + \alpha_2 \underline{y}_{t+\tau-2} + \alpha_1 \underline{y}_{t+\tau-1} + \underline{n}_{t+\tau}, \quad (73.53)$$

we get

$$Z_{(t,t+\tau)} \underline{y}_{t+\tau} = \underline{n}_{t+\tau} \quad \text{if } \tau > p. \quad (73.54)$$

Hence, for $\tau > p$,

$$\gamma^{part}(\tau) = \left\langle \underline{y}_t \underline{n}_{t+\tau} \right\rangle = 0 \quad (73.55)$$

QED

Fig.73.5 gives a graphical proof, using bnets and the d-separation theorem, that for an AR(2) t-series, $\gamma^{part}(\tau) = 0$ for $\tau > 2$. As a consequence of this, a plot of $\gamma^{part}(\tau)$ versus τ for a typical AR(2) t-series looks like Fig.73.6.

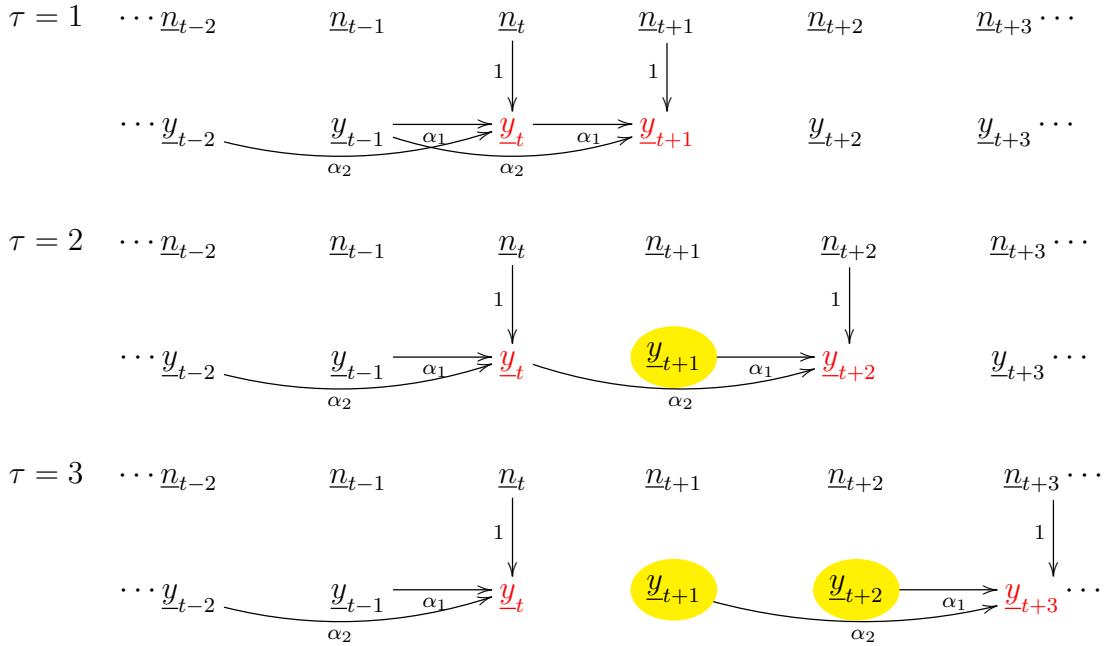


Figure 73.5: $\text{AR}(2)$ bnet. For clarity, we show only arrows entering nodes y_t and $y_{t+\tau}$ for $\tau = 1, 2, 3$. Yellow nodes are in set $y_{(t,t+\tau)}$. They are conditioned on, so information can't flow through them to node $y_{t+\tau}$ by the d-separation theorem.

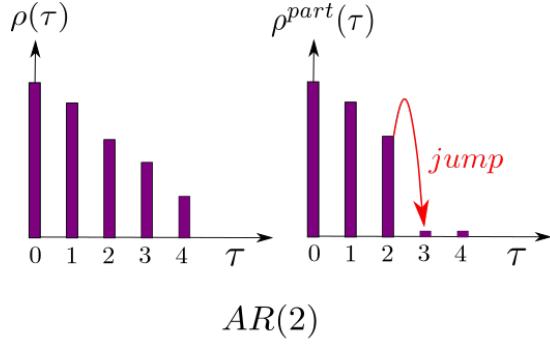


Figure 73.6: Plot of auto-correlation function (ACorrF) $\rho(\tau)$ and partial auto-correlation function (PACorrF) $\rho^{\text{part}}(\tau)$ for an instance of $\text{AR}(2)$. Note that $\rho^{\text{part}}(\tau)$ vanishes for $\tau > 2$.

	ACorr	PACorr
$AR(p)$	tapers off	jumps to zero for $\tau > p$
$MA(q)$	jumps to zero for $\tau > q$	tapers off

Table 73.1: Detecting $AR(p)$ and $MA(q)$ using auto-correlation and partial auto-correlation.

73.9 Generating function of auto-correlation

Claim 99 *If*

$$\alpha(z) = \sum_{\tau=-\infty}^{\infty} \alpha_{\tau} z^{\tau} \quad (73.56)$$

then

$$\alpha(z)\alpha(z^{-1}) = \sum_{\tau=-\infty}^{\infty} z^{\tau} \sum_{j=-\infty}^{\infty} \alpha_j \alpha_{j+\tau} \quad (73.57)$$

proof:

$$\alpha(z)\alpha(z^{-1}) = \sum_{j'=-\infty}^{\infty} z^{j'} \alpha_{j'} \sum_{j=-\infty}^{\infty} z^{-j} \alpha_j \quad (73.58)$$

$$= \sum_{\tau=-\infty}^{\infty} \sum_{j'=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} z^{\tau} \alpha_{j'} \alpha_j \delta(j' - j, \tau) \quad (73.59)$$

$$= \sum_{\tau=-\infty}^{\infty} z^{\tau} \sum_{j=-\infty}^{\infty} \alpha_j \alpha_{j+\tau} \quad (73.60)$$

QED

As an example of this claim, note that

$$(\alpha_0 + \alpha_1 z)(\alpha_0 + \alpha_1 z^{-1}) = \alpha_0 \alpha_1 z^{-1} + (\alpha_0^2 + \alpha_1^2) + \alpha_0 \alpha_1 z \quad (73.61)$$

Claim 100 *If $\{\underline{y}_t\}_{\forall t}$ is an $AR(p)$ t-series, then*

$$\tilde{\gamma}(z) = \alpha'(z) \sigma^2 \alpha'(z^{-1}) \quad (73.62)$$

proof: Left to reader. See Claim 99.

QED

Claim 101 *If $\{\underline{y}_t\}_{\forall t}$ is an $MA(q)$ t-series, then*

$$\tilde{\gamma}(z) = \nu(z) \sigma^2 \nu(z^{-1}) \quad (73.63)$$

proof:

$$\underline{y}_t = \sum_{j=-\infty}^{\infty} \nu_j \underline{n}_{t-j} \quad (73.64)$$

$$\tilde{\gamma}(z) = \sum_{\tau=-\infty}^{\infty} \left\langle \underline{y}_0, \underline{y}_{\tau} \right\rangle z^{\tau} \quad (73.65)$$

$$= \sum_{\tau=-\infty}^{\infty} z^{\tau} \sum_{j=-\infty}^{\infty} \sum_{j'=-\infty}^{\infty} \nu_j \nu_{j'} \underbrace{\left\langle \underline{n}_{-j}, \underline{n}_{\tau-j'} \right\rangle}_{\sigma^2 \delta(j', \tau+j)} \quad (73.66)$$

$$= \sigma^2 \sum_{\tau=-\infty}^{\infty} z^{\tau} \sum_{j=-\infty}^{\infty} \nu_j \nu_{j+\tau} \quad (73.67)$$

Now use Claim 99.

QED

Claim 102 *If $\{\underline{y}_t\}_{\forall t}$ is an ARMA(p, q) t-series, then*

$$\tilde{\gamma}(z) = \alpha'(z) \nu(z) \sigma^2 \nu(z^{-1}) \alpha'(z^{-1}) \quad (73.68)$$

proof: Left to reader. See Claim 99.

QED

73.10 Impulse Response

The derivatives

$$IR_{\tau} = \frac{\partial y_{t+\tau}}{\partial n_t} \quad (73.69)$$

are called **impulse responses** or **dynamical multipliers**. Note that this derivative depends on τ but not t by w-stationarity. A plot of IR_{τ} versus τ is called the **Impulse Response Function (IRF)**. Examples:

- **Claim 103** *For MA(q),*

$$\frac{\partial y_{t+\tau}}{\partial n_t} = \nu_{\tau} \mathbb{1}(0 \leq \tau \leq q) \quad (73.70)$$

where $\nu_0 = 1$. (See Fig. 73.7)

proof:

$$\underline{y}_{t+\tau} = \sum_{j=0}^q \nu_j \underline{n}_{t+\tau-j} \quad (73.71)$$

so Eq.(73.70) follows.

QED

- **Claim 104** *For AR(1),*

$$\frac{\partial y_{t+\tau}}{\partial n_t} = (\alpha_1)^\tau \mathbb{1}(\tau \geq 0). \quad (73.72)$$

(See Fig. 73.8)

proof:

$$(1 - \alpha_1 \mathcal{B}) \underline{y}_{t+\tau} = \underline{n}_{t+\tau} \quad (73.73)$$

Therefore, using the Taylor expansion Eq.(73.7),

$$\underline{y}_{t+\tau} = 1 + \sum_{j=1}^{\infty} \alpha_1^j \mathcal{B}^j \underline{n}_{t+\tau} \quad (73.74)$$

$$= 1 + \sum_{j=1}^{\infty} \alpha_1^j \underline{n}_{t+\tau-j} \quad (73.75)$$

so Eq.(73.72) follows.

QED

In general, IR_τ equals a sum over paths from \underline{n}_t to $\underline{y}_{t+\tau}$. Each path contributes the product of the weights of all the arrows in the path.

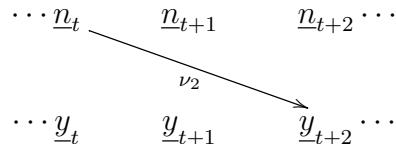


Figure 73.7: Pictorial representation of impulse response $IR_2 = \nu_2$ for a $MA(q)$ t-series with $q \geq 2$.

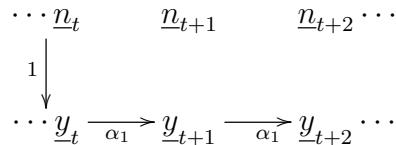


Figure 73.8: Pictorial representation of impulse response $IR_2 = \alpha_1^2$ for an $AR(1)$ t-series.

73.11 AR(p) and Yule-Walker equations

Claim 105 (*Yule-Walker equations*) If $\{\underline{y}_t\}_{\forall t}$ is an AR(p) t-series,

$$\gamma_\tau = \sum_{j=1}^p \gamma_{\tau-j} \alpha_j + \sigma^2 \delta(\tau, 0) \quad (73.76)$$

for all $\tau \in \mathbb{Z}$, where we are abbreviating $\gamma(j) = \gamma_j$ for all j .

proof:

Recall that

$$\underline{y}_\tau = \sum_{j=1}^p \underline{y}_{\tau-j} \alpha_j + \underline{n}_\tau \quad (73.77)$$

Therefore

$$\underbrace{\langle \underline{y}_0, \underline{y}_\tau \rangle}_{\gamma_\tau} = \sum_{j=1}^p \underbrace{\langle \underline{y}_0, \underline{y}_{\tau-j} \rangle}_{\gamma_{\tau-j}} \alpha_j + \underbrace{\langle \underline{y}_0, \underline{n}_\tau \rangle}_{\sigma^2 \delta(\tau, 0)} \quad (73.78)$$

QED

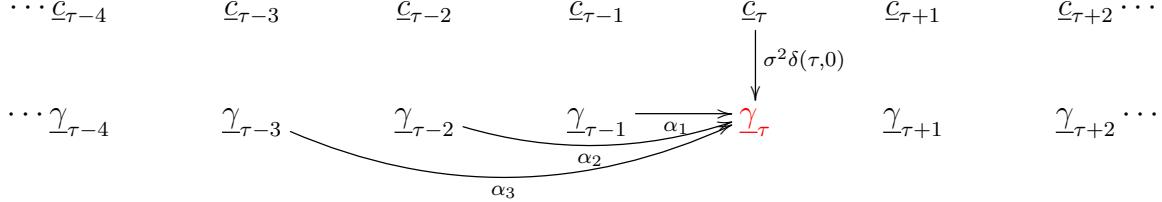


Figure 73.9: Bnet representing Yule-Walker (Christmas Walker) Eqs.(73.76) with $p = 3$. For clarity, we show only arrows entering node γ_τ . The full bnet has the same structural pattern of incoming arrows (including the α_j) for each node γ_τ' for all τ' .

Fig.73.9 shows a bnet for the Yule-Walker Eqs.(73.76). The TPMs, printed in blue, for the τ time slice of that bnet, are as follows:

$$P(\gamma_\tau) = \mathbb{1} (\gamma_\tau = \text{ See Eq.(73.76)}) \quad (73.79)$$

$$P(c_\tau) = \mathbb{1}(c_\tau = \sigma^2 \delta(\tau, 0)) \quad (73.80)$$

The Yule-Walker Eqs.(73.76) imply that

$$\underbrace{\begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_p \end{bmatrix}}_{\gamma^p} = \underbrace{\begin{bmatrix} \gamma_0 & \gamma_{-1} & \gamma_{-2} & \cdots & \gamma_{1-p} \\ \gamma_1 & \gamma_0 & \gamma_{-1} & \cdots & \gamma_{2-p} \\ \vdots & & & & \\ \gamma_{p-1} & \gamma_{p-2} & \gamma_{p-3} & \cdots & \gamma_0 \end{bmatrix}}_{\Gamma} \underbrace{\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_p \end{bmatrix}}_{\alpha^p} \quad (73.81)$$

Hence

$$\gamma^p = \Gamma \alpha^p . \quad (73.82)$$

If Γ is invertible,

$$\alpha^p = \Gamma^{-1} \gamma^p . \quad (73.83)$$

Once we know α^p , we can solve for σ^2

$$\gamma_0 = \sum_{j=1}^p \gamma_{-j} \alpha_j + \sigma^2 , \quad (73.84)$$

i.e., Eq.(73.76) for $\tau = 0$.

73.12 Forecasting

Before submerging the reader in the messy details of t-series forecasting, and running the risk of quickly losing him/her, let me explain to the reader that what we are about to do in this section, is, in the final analysis, quite trivial, and boils down to solving simple systems of linear equations. That is all there is to it!! How hard can that be? In reading this section, don't lose sight of that fact and you'll be okay.

Suppose $\{\underline{y}_t\}_{\forall t}$ is an $AR(\infty)$ t-series. For $\tau > 0$, the “orthogonal projection” $\underline{y}_{t+\tau} | \underline{y}_{\leq t}$ is defined as the linear combination of $\underline{y}_{\leq t}$ obtained by doing Linear Regression with x-variables $\underline{y}_{\leq t}$ and y-variable $\underline{y}_{t+\tau}$. Thus

$$E \left[\underline{y}_{t+\tau} | \underline{y}_{\leq t} \right] = E_{|\underline{y}_{\leq t}} [\underline{y}_{t+\tau}] \quad (73.85)$$

and

$$Var \left[\underline{y}_{t+\tau} | \underline{y}_{\leq t} \right] = E \left[\left(\underline{y}_{t+\tau} | \underline{y}_{\leq t} - E[\underline{y}_{t+\tau} | \underline{y}_{\leq t}] \right)^2 \right] \quad (73.86)$$

$$= Var_{|\underline{y}_{\leq t}} [\underline{y}_{t+\tau}] . \quad (73.87)$$

For example, for $\tau = 1$,

$$\underline{y}_{t+1} = \sum_{j=1}^p \alpha_j \underline{y}_{t+1-j} + \underline{n}_{t+1} \quad (73.88)$$

$$\underline{y}_{t+1} | \underline{y}_{\leq t} = \sum_{j=1}^p \alpha_j \underline{y}_{t+1-j} \quad (73.89)$$

$$\underline{y}_{t+1} - \underline{y}_{t+1 | \underline{y}_{\leq t}} = \underline{n}_{t+1} \quad (73.90)$$

$$E \left[\underline{y}_{t+1} | \underline{y}_{\leq t} \right] = E_{|\underline{y}_{\leq t}} [\underline{y}_{t+1}] \quad (73.91)$$

$$= 0 \quad (73.92)$$

$$Var \left[\underline{y}_{t+1} | \underline{y}_{\leq t} \right] = Var_{|\underline{y}_{\leq t}} [\underline{y}_{t+1}] \quad (73.93)$$

$$= \sigma^2 \quad (73.94)$$

If $\{\underline{y}_t\}_{\forall t}$ is an $AR(\infty)$ t-series and $\tau > 0$, by **forecasting** $\underline{y}_{t+\tau} | \underline{y}_{\leq t}$, we will mean finding $\underline{y}_{t+\tau} | \underline{y}_{\leq t}$, given $\underline{y}_{\leq t}$ as input data or some other input data from which $\underline{y}_{\leq t}$ can be derived. Fig.73.10 illustrates 3 types of input data that we will consider next.

Note that if $\{\underline{y}_t\}_{\forall t}$ is an $AR(\infty)$ t-series, then $\alpha^-(\mathcal{B})\underline{y}_t = \underline{n}_t$. Thus, $\underline{n}_{\leq t}$ can be expressed in terms of $\underline{y}_{\leq t}$, yielding a function $\underline{n}_{\leq t}(\underline{y}_{\leq t})$. If the operator $\alpha^-(\mathcal{B})$ is invertible, the opposite is also true: $\underline{y}_{\leq t}$ can be expressed in terms of $\underline{n}_{\leq t}$, yielding a function $\underline{y}_{\leq t}(\underline{n}_{\leq t})$. Thus,

$$\underline{y}_{t+\tau} | \underline{y}_{\leq t} = \underline{y}_{t+\tau} | \underline{y}_{\leq t}(\underline{n}_{\leq t}) = \underline{y}_{t+\tau} | \underline{n}_{\leq t} \quad (73.95)$$

We've seen that if we assume invertibility of $\alpha^-(\mathcal{B})$, then, in order to forecast $\underline{y}_{t+\tau} | \underline{y}_{\leq t}$, we have the option of considering $\{\underline{y}_t\}_{\forall t}$ to be either an $AR(\infty)$ or an $MA(\infty)$ t-series. We will assume the latter, because this simplifies the algebra necessary to calculate both $\underline{y}_{t+\tau} | \underline{y}_{\leq t}$ and its variance.

(a) **find** $\underline{y}_{t+\tau} | \underline{y}_{\leq t}$ **given** $\underline{y}_{\leq t}$

Suppose $\{\underline{y}_t\}_{\forall t}$ is an $MA(\infty)$ t-series. Then

$$\underline{y}_{t+\tau} = \sum_{j=0}^{\infty} \nu_j \underline{n}_{t+\tau-j} \quad (73.96)$$

$$= \underbrace{\sum_{j=0}^{\tau-1} \nu_j \underline{n}_{t+\tau-j}}_{\underline{y}_{t+\tau} - \underline{y}_{t+\tau} | \underline{y}_{\leq t}} + \underbrace{\sum_{j=\tau}^{\infty} \nu_j \underline{n}_{t+\tau-j}}_{\underline{y}_{t+\tau} | \underline{y}_{\leq t}} \quad (73.97)$$

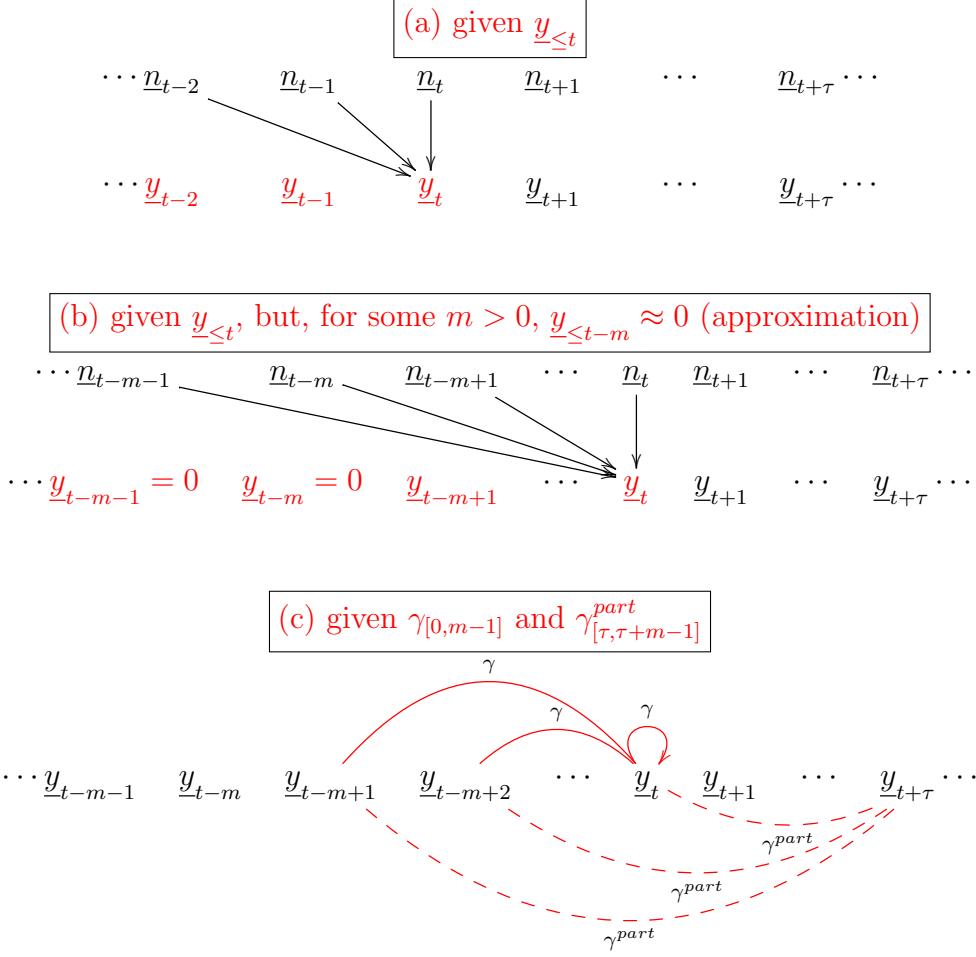


Figure 73.10: Bnet for $MA(\infty)$ for forecasting $\underline{y}_{t+\tau} | \underline{y}_{\leq t}$. Input data in red. For clarity, we show only the arrows entering node \underline{y}_t .

$$\underline{y}_{t+\tau} | \underline{y}_{\leq t} = \sum_{j=\tau}^{\infty} \nu_j \underline{n}_{t+\tau-j} \quad (73.98)$$

$$E_{|\underline{y}_{\leq t}}[\underline{y}_{t+\tau}] = E[\underline{y}_{t+\tau} | \underline{y}_{\leq t}] = 0^1 \quad (73.99)$$

$$\left\langle \underline{y}_{t+\tau} \underline{y}_{t+\tau} \right\rangle_{|\underline{y}_{\leq t}} = \left\langle \underline{y}_{t+\tau} - \underline{y}_{t+\tau} | \underline{y}_{\leq t}, \underline{y}_{t+\tau} - \underline{y}_{t+\tau} | \underline{y}_{\leq t} \right\rangle \quad (73.100)$$

$$= \sum_{j=0}^{\tau-1} \nu_j^2 \quad (73.101)$$

¹Remember that \underline{y}_t has zero mean by definition. $Y_t = y_t + \mu$ for some time independent constant μ .

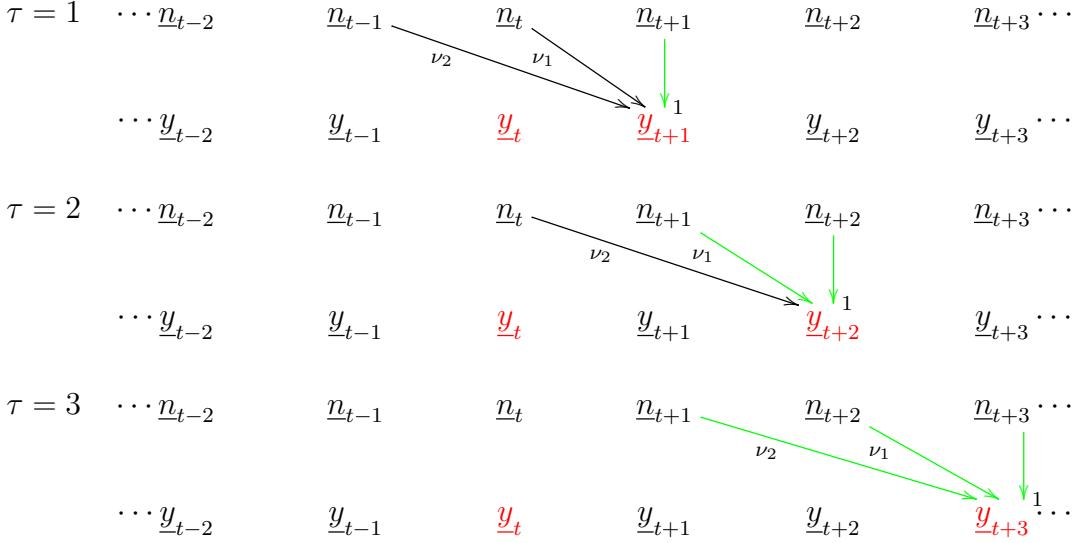


Figure 73.11: $MA(2)$ bnet. For clarity, we show only arrows entering node $\underline{y}_{t+\tau}$ for $\tau = 1, 2, 3$. Green arrows originate at a time after time t .

For $MA(q)$, because $\nu_j = 0$ for $j > q$, this reduces to

$$\underline{y}_{t+\tau} | y_{\leq t} = \mathbb{1}(\tau \leq q) \sum_{j=\tau}^q \nu_j \underline{n}_{t+\tau-j} \quad (73.102)$$

$$E_{|y_{\leq t}}[\underline{y}_{t+\tau}] = E[\underline{y}_{t+\tau} | y_{\leq t}] = 0 \quad (73.103)$$

$$\left\langle \underline{y}_{t+\tau}, \underline{y}_{t+\tau} \right\rangle_{|y_{\leq t}} = \sum_{j=0}^{\min(\tau-1, q)} \nu_j^2 \quad (73.104)$$

Eq.(73.104) is motivated by Fig.73.11). Only the green arrows entering $\underline{y}_{t+\tau}$ and originating in a node $\underline{n}_{t'}$ with $t' > t$ contribute to the right hand side of Eq.(73.104).

Thus, the mean of $\underline{y}_{t+\tau}$ can be predicted to remain zero for all $\tau > 0$ (forever). The error in that prediction increases with τ until τ reaches q . After that, the error remains constant.

Note that

$$\underline{y}_{t+\tau} | y_{\leq t} = \sum_{j=\tau}^{\infty} \nu_j \underline{n}_{t+\tau-j} \quad (73.105)$$

$$= \sum_{j=\tau}^{\infty} \nu_j \mathcal{B}^{j-\tau} \underline{n}_t \quad (73.106)$$

$$= \left[\sum_{j=0}^{\infty} \nu_j \mathcal{B}^{j-\tau} \right]_{\mathcal{B} \geq 0} \underline{n}_t \quad (73.107)$$

$$= [\mathcal{B}^{-\tau} \nu(\mathcal{B})]_{\mathcal{B} \geq 0} \underline{n}_t \quad (73.108)$$

$\mathcal{B} \geq 0$ means only the non-negative powers of \mathcal{B} are kept. Eq.(73.108) is just Eq.(73.98) in fancy notation.

In $MA(\infty)$, if $\nu(\mathcal{B})$ is invertible, then

$$\underline{n}_t = \nu(\mathcal{B})^{-1} \underline{y}_t \quad (73.109)$$

so

$$\boxed{\underline{y}_{t+\tau} | y_{\leq t} = \mathcal{L}_{WK} \underline{y}_t} \quad (73.110)$$

where

$$\mathcal{L}_{WK} = [\mathcal{B}^{-\tau} \nu(\mathcal{B})]_{\mathcal{B} \geq 0} \nu(\mathcal{B})^{-1} \quad (73.111)$$

Eq.(73.110) is called the **Wiener-Kolgomorov prediction formula (WKPF)**. Next, we apply WKPF to $AR(1)$ and $MA(1)$ as examples of its usage.

- $AR(1)$

$$\underbrace{(1 - \alpha_1 \mathcal{B})}_{\nu(\mathcal{B})^{-1}} \underline{y}_t = \underline{n}_t \quad (73.112)$$

$$\underline{y}_t = \underbrace{\left[\sum_{j=0}^{\infty} (\alpha_1 \mathcal{B})^j \right]}_{\nu(\mathcal{B})} \underline{n}_t \quad (73.113)$$

$$\nu_j = (\alpha_1)^j \quad (73.114)$$

$$[\mathcal{B}^{-\tau} \nu(\mathcal{B})]_{\mathcal{B} \geq 0} = (\alpha_1)^{\tau} \nu(\mathcal{B}) \quad (73.115)$$

$$\underline{y}_{t+\tau} | y_{\leq t} = (\alpha_1)^{\tau} \underline{y}_t \quad (73.116)$$

Hence, $\underline{y}_{t+\tau} | y_{\leq t}$ decreases geometrically as τ grows.

- $MA(1)$

$$\underline{y}_t = \underbrace{(1 + \nu_1 \mathcal{B})}_{\nu(\mathcal{B})} \underline{n}_t \quad (73.117)$$

$$[\mathcal{B}^{-\tau} \nu(\mathcal{B})]_{\mathcal{B} \geq 0} = \delta(\tau, 1) \nu_1 \quad (73.118)$$

$$\underline{y}_{t+\tau} | \underline{y}_{\leq t} = \delta(\tau, 1) \nu_1 \underbrace{\left[\sum_{j=0}^{\infty} (-\nu_1 \mathcal{B})^j \right]}_{\nu(\mathcal{B})^{-1}} \underline{y}_t \quad (73.119)$$

$$= \delta(\tau, 1) \nu_1 \underline{n}_t \quad (73.120)$$

(b) find $\underline{y}_{t+\tau} | \underline{y}_{\leq t}$ given $\underline{y}_{\leq t}$, but, for some $m > 0$, $\underline{y}_{\leq t-m} \approx 0$ (approximation)

$$\underline{y}_{t+\tau} | \underline{y}_{\leq \tau} \approx [\mathcal{L}_{WK}]_{\mathcal{B}^{<m}} \underline{y}_t \quad (73.121)$$

(c) find $\underline{y}_{t+\tau} | \underline{y}_{\leq t}$ given $\gamma_{[0, m-1]}$ and $\gamma_{[\tau, \tau+m-1]}^{part}$

$$\underline{y}_{t+\tau} | \underline{y}_{\leq t} = \sum_{j=0}^{m-1} \underline{y}_{t-j} \beta_j \quad (73.122)$$

Suppose that $k \in \mathbb{Z}_{[0, m-1]}$.

Note that

$$\underline{y}_{t-k} | \underline{y}_{\leq t} = \underline{y}_{t-k} . \quad (73.123)$$

$$\underbrace{\langle \underline{y}_{t-k} | \underline{y}_{\leq t}, \underline{y}_{t+\tau} | \underline{y}_{\leq t} \rangle}_{\gamma_{\tau+k}^{part}} = \sum_{j=0}^{m-1} \underbrace{\langle \underline{y}_{t-k}, \underline{y}_{t-j} \rangle}_{\gamma_{k-j}} \beta_j \quad (73.124)$$

$$\begin{bmatrix} \gamma_{\tau}^{part} \\ \gamma_{\tau+1}^{part} \\ \vdots \\ \gamma_{\tau+m-1}^{part} \end{bmatrix} = \underbrace{\begin{bmatrix} \gamma_0 & \gamma_{-1} & \gamma_{-2} & \dots & \gamma_{1-m} \\ \gamma_1 & \gamma_0 & \gamma_{-1} & \dots & \gamma_{2-m} \\ \vdots & & & & \\ \gamma_{m-1} & \gamma_{m-2} & \gamma_{m-3} & \dots & \gamma_0 \end{bmatrix}}_{\Gamma} \underbrace{\begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{m-1} \end{bmatrix}}_{\beta^m} \quad (73.125)$$

Solve Eq.(73.125) for β^m and plug β^m into Eq.(73.122).

73.13 Model Learning

Box-Jenkins Method

1. Filtering Data (FD)

Removing trends and periodicity (seasonality) via differencing, so as to achieve a w-stationary t-series. We deal with FD in Section 73.14.

2. Model Selection

Finding best possible p, q for $ARMA(p, q)$ using various statistical tests and metrics.

3. Parameter Learning (PL)

Finding best possible coefficients α_j and ν_j . We deal with PL in Section 73.15.

4. Testing

Measuring the goodness of fit.

73.14 Differencing and $ARIMA(p, d, q)$

Let $\alpha \in (0, 1)$. We say that $\{\underline{s}_t\}_{t \in \mathbb{N}}$ is a **Simple Exponential Smoothing (SES) t-series** if it satisfies

$$s_t = (1 - \alpha)s_{t-1} + \alpha x_t \quad (73.126)$$

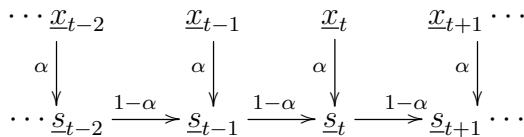


Figure 73.12: Bnet for Simple Exponential Smoothing t-series. $\alpha \in (0, 1)$

A SES t-series can be represented by the bnet of Fig.73.12. The TPM, printed in blue, for node s_t , is as follows

$$P(s_t) = \mathbb{1}(s_t = \text{ see Eq.(73.126)}) \quad (73.127)$$

One has

$$(1 - (1 - \alpha)\mathcal{B})\underline{s}_t = \alpha\underline{x}_t \quad (73.128)$$

so

$$\underline{s}_t = \frac{\alpha}{1 - (1 - \alpha)\mathcal{B}} \underline{x}_t \quad (73.129)$$

$$= \alpha \sum_{j=0}^{\infty} (1 - \alpha)^j \mathcal{B}^j \underline{x}_t \quad (\text{by Eq.(73.7)}) \quad (73.130)$$

Note from Fig.73.12 and Eq.(73.130) that each shock \underline{x}_{t-j} with $j > 0$ contributes to \underline{s}_t proportionally to the product $\alpha(1 - \alpha)^j$ of arrow weights in the path from \underline{x}_{t-j} to \underline{s}_t . For example, \underline{x}_{t-2} contributes to \underline{s}_t in the proportion $\alpha(1 - \alpha)^2$. The contributions of \underline{x}_{t-j} to \underline{s}_t decrease geometrically as the lag j increases.

Henceforth, we will abbreviate “differencing” by “diff”.

- First order diff operator

$$\Delta x_t = x_t - x_{t-1} = (1 - \mathcal{B})x_t \quad (73.131)$$

Seasonal first order diff:

$$\Delta_{sea} x_t = x_t - x_{t-ts} \quad (73.132)$$

where ts is length of season.

- Second order diff operator

$$\Delta^2 x_t = \Delta x_t - \Delta x_{t-1} \quad (73.133)$$

$$= x_t - 2x_{t-1} + x_{t-2} \quad (73.134)$$

$$= (1 - 2\mathcal{B} + \mathcal{B}^2)x_t \quad (73.135)$$

$$= (1 - \mathcal{B})^2 x_t \quad (73.136)$$

- d -th order diff operator

$$\Delta^d \underline{x}_t = (1 - \mathcal{B})^d \underline{x}_t \quad (73.137)$$

Recall that if $\{\underline{y}_t\}_{\forall t}$ is an $ARMA(p + d, q)$ t-series,

$$\left(1 - \sum_{j=1}^{p+d} \alpha_j \mathcal{B}^j\right) \underline{y}_t = \left(1 + \sum_{k=1}^q \nu_k \mathcal{B}^k\right) \underline{n}_t \quad (73.138)$$

We will say that $\{\underline{y}_t\}_{\forall t}$ is an **Autoregressive Integrated Moving Average $ARIMA(p, d, q)$ t-series** if

$$\left(1 - \sum_{j=1}^p \alpha_j \mathcal{B}^j\right) \underbrace{\Delta^d \underline{y}_t}_{(1-\mathcal{B})^d \underline{y}_t} = \left(1 + \sum_{k=1}^q \nu_k \mathcal{B}^k\right) \underline{n}_t \quad (73.139)$$

So an $ARIMA(p, d, q)$ t-series is a special case of an $ARMA(p + d, q)$ t-series.

Define \mathcal{O}_{BEF} and \mathcal{O}_{AFT} by

$$\mathcal{O}_{BEF} = 1 - \alpha_1 \mathcal{B} - \alpha_2 \mathcal{B}^2 - \alpha_3 \mathcal{B}^3 \quad (73.140)$$

$$\mathcal{O}_{AFT} = \mathcal{O}_{BEF}(1 - \mathcal{B}) \quad (73.141)$$

$$= \left\{ \begin{array}{cccc} 1 & -\alpha_1 \mathcal{B} & -\alpha_2 \mathcal{B}^2 & -\alpha_3 \mathcal{B}^3 \\ & -\mathcal{B} & +\alpha_1 \mathcal{B}^2 & +\alpha_2 \mathcal{B}^3 \\ \end{array} \right. + \alpha_3 \mathcal{B}^4 \quad (73.142)$$

$$= 1 - \underbrace{(\alpha_1 + 1)}_{\alpha'_1} \mathcal{B} - \underbrace{(\alpha_2 - \alpha_1)}_{\alpha'_2} \mathcal{B}^2 - \underbrace{(\alpha_3 - \alpha_2)}_{\alpha'_3} \mathcal{B}^3 - \underbrace{(0 - \alpha_3)}_{\alpha'_4} \mathcal{B}^4 \quad (73.143)$$

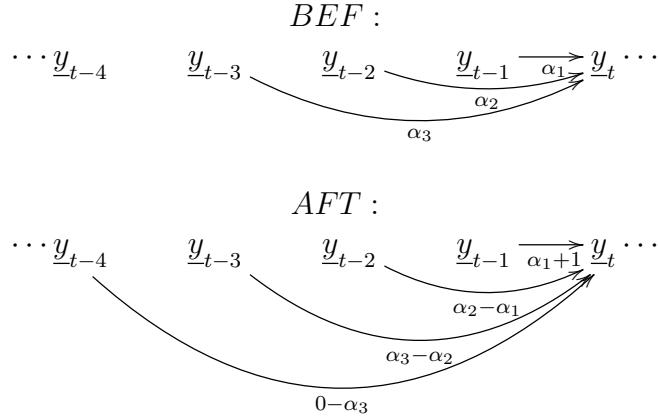


Figure 73.13: Effect on y_t of going from before (BEF) with a t-series $ARIMA(3,0,0)$ to after (AFT) with a t-series $ARIMA(3,1,0)$.

$\mathcal{O}_{BEF}y_t$ and $\mathcal{O}_{AFT}y_t$ are illustrated in Fig.73.13. Note that in going from BEF to AFT, a new arrow is introduced with weight $\alpha'_4 = -\alpha_3$. The intermediate length arrows $y_{t-3} \rightarrow y_t$ and $y_{t-2} \rightarrow y_t$ have weights $\alpha'_3 = \alpha_3 - \alpha_2$ and $\alpha'_2 = \alpha_2 - \alpha_1$ so if $\alpha_j \approx \alpha_{j-1}$, then those 2 arrows have negligible weights. Only the the longest arrow $y_{t-4} \rightarrow y_t$ and the shortest arrow $y_{t-1} \rightarrow y_t$ have non-negligible weights.

Suppose

$$y_t \approx a + bt + ct^2 + dt^3 \quad (73.144)$$

This smooth, low-order polynomial-fit to the t-series is called its **trend**.

$$\Delta y_t \approx \Delta t \frac{d}{dt} y_t = b + 2ct + 3dt^2 \quad (73.145)$$

$$\Delta^2 y_t \approx (\Delta t)^2 \frac{d^2}{dt^2} y_t = 2c + 6dt \quad (73.146)$$

$$\Delta^3 \underline{y}_t \approx (\Delta t)^3 \frac{d^2}{dt^3} \underline{y}_t = 6d \quad (73.147)$$

Note that the mean $6d$ in $\Delta^3 \underline{y}_t$ can be adjusted to zero (demeaning). From this example, we see that every time we apply a diff operator to a time series, we reduce the order of its polynomial trend by one.

Note that if $\{\underline{y}_t\}_{\forall t}$ is an $ARIMA(0, 1, 1)$ t-series,

$$\Delta \underline{y}_t = (1 + \nu_1 \mathcal{B}) \underline{n}_t \quad (73.148)$$

$$\underline{y}_t = \underline{y}_{t-1} + \underline{n}_t + \nu_1 \underline{n}_{t-1} \quad (73.149)$$

Claim 106 $ARIMA(0, 1, 1)$ is equivalent to a simple exponential smoothing t-series.

proof: Setting

$$\underline{n}_t = \underline{y}_t - \hat{\underline{y}}_t \quad (73.150)$$

in Eq.(73.148), we get

$$(1 - \mathcal{B}) \underline{y}_t = (1 - \nu_1 \mathcal{B}) (\underline{y}_t - \hat{\underline{y}}_t) \quad (73.151)$$

$$(1 - \nu_1 \mathcal{B}) \hat{\underline{y}}_t = (1 - \nu_1) \mathcal{B} \underline{y}_t \quad (73.152)$$

$$\hat{\underline{y}}_t = \nu_1 \hat{\underline{y}}_{t-1} + (1 - \nu_1) \underline{y}_{t-1} \quad (73.153)$$

Now replace $\hat{\underline{y}}_t \rightarrow \underline{s}_t$, $\underline{y}_{t-1} \rightarrow \underline{n}_t$ and $\nu_1 \rightarrow 1 - \alpha$.

Note that this proof has established the equivalence of two different bnets. Those two bnets are pictured in Fig.73.14.

QED

73.15 Parameter Learning

In this section, we will show how to find an estimate $\hat{\theta}$ for the parameters $\theta = (\{\alpha_j\}_{\forall j}, \{\nu_j\}_{\forall j}, \sigma^2)$ used in $ARMA(p, q)$. We do this **parameter learning (PL)** by postulating a “quasi” (log-) likelihood function $\mathcal{L}(\theta)$, and maximizing that over θ . The estimate $\hat{\theta}$ that we obtain is called the **Quasi-Maximum Likelihood Estimate (QMLE)**. The reason for using the prefix “quasi” is that the likelihood probability involved doesn’t really satisfy the i.i.d assumption.

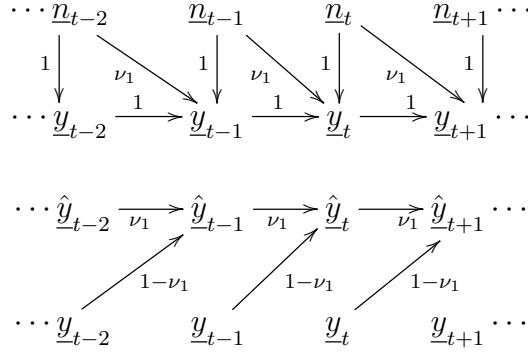


Figure 73.14: Bnets for two t-series that were proven equivalent in Claim 106. Note that the 2 bnets have NO arrows in common!

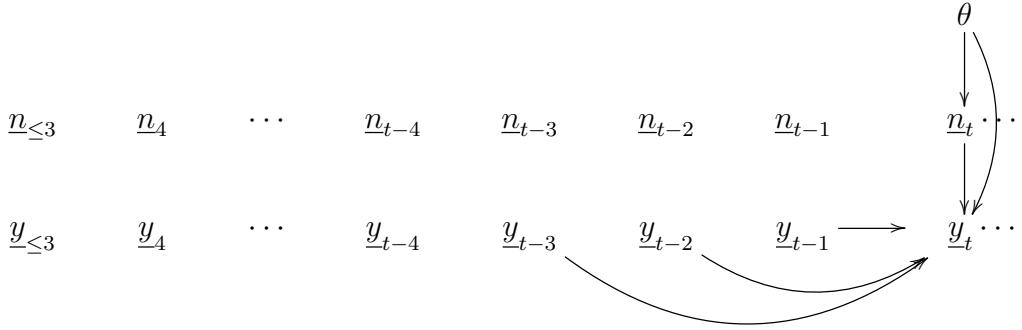


Figure 73.15: Bnet for PL of $AR(3)$. For clarity, we show only the arrows entering nodes y_t and n_t .

73.15.1 PL of $AR(p)$

Recall that if $\{y_t\}_{\forall t}$ is an $AR(p)$ t-series,

$$y_t = \sum_{j=1}^p \alpha_j y_{t-j} + n_t \quad (73.154)$$

Let $\theta = (\sigma^2, \alpha_{[1,p]})$. We want to learn the parameters θ of the $AR(p)$ bnet Fig.73.15, assuming the TPMs, printed in blue, are as follows.

For $t \geq p + 1$

$$P(y_t | y_{<t}, n_t, \theta) = \mathbb{1}(y_t = \text{see Eq.(73.154)}) \quad (73.155)$$

$$P(n_t | \theta) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[\frac{-(n_t)^2}{2\sigma^2}\right] \quad (73.156)$$

Note that

$$P(y_{\leq t}|\theta) = \sum_{n_{[p+1,t]}} \left\{ \prod_{\tau \in [p+1,t]} P(y_\tau|y_{[\tau-p,\tau-1]}, n_\tau) P(n_\tau|\theta) \right\} P(y_{\leq p}|\theta) \quad (73.157)$$

$$= \prod_{\tau \in [p+1,t]} \left\{ \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left[\frac{-\left(y_\tau - \sum_{j=1}^p \alpha_j y_{\tau-j}\right)^2}{2\sigma^2} \right] \right\} P(y_{\leq p}|\theta) \quad (73.158)$$

The log likelihood function $\mathcal{L}_t(\theta)$ for this bnet is defined by

$$\mathcal{L}_t(\theta) = \frac{1}{t} \ln P(y_{\leq t}|\theta). \quad (73.159)$$

Hence,

$$\mathcal{L}_t(\theta) = \underbrace{\frac{1}{t} \ln P(y_{\leq p}|\theta)}_{\mathcal{L}_t^{prior}(\theta)} + \mathcal{L}'_t(\theta) \quad (73.160a)$$

where

$$\mathcal{L}'_t(\theta) = \frac{-(t-p)}{t} \ln \sqrt{2\pi\sigma^2} - \frac{1}{2t\sigma^2} \sum_{\tau=p+1}^t \left(y_\tau - \sum_{j=1}^p \alpha_j y_{\tau-j} \right)^2. \quad (73.160b)$$

Henceforth, we will assume that $y_{[1,t]}$ is known.

We start by assuming that the prior log-likelihood $\mathcal{L}_t^{prior}(\theta)$ is zero. This means that we have uniform (uninformative) priors. Later on, we will consider instead a Gaussian prior that uses the info that $y_{\leq p}$ is known a priori.

If we assume that $y_{[1,t]}$ is known, then we can plug this info into $\mathcal{L}'_t(\theta)$ and use a non-linear optimization method to maximize $\mathcal{L}'_t(\theta)$ and find the optimum θ .

Alternatively, one can use LR. Define the strange looking dataset

$$\mathcal{D} = \{(t, y_{[t-p,t-1]}, \boxed{y_t}) : t\} \quad (73.161)$$

This dataset is strange because it replaces sampling from a population with sampling in time. (i.e., a “stochastic” average by an “ergodic” average.) The datasets that we use to do LR usually satisfy the i.i.d. assumption, but that assumption does not necessarily hold for this one. Luckily, that assumption is not necessary for doing LR with x-variables $y_{[t-p,t-1]}$, y-variable y_t , and regression coefficients $\alpha_{[1,p]}$. The LR software gives estimates $\hat{\alpha}_{[1,p]}$ that we can use to calculate residuals

$$\epsilon_\tau = y_\tau - \sum_{j=1}^p \hat{\alpha}_j y_{\tau-j} \quad (73.162)$$

One can estimate σ^2 from those residuals using

$$\widehat{\sigma}^2 = \frac{1}{t-p} \sum_{\tau=p+1}^t \epsilon_\tau^2. \quad (73.163)$$

So far, we have assumed an uninformative prior. Alternatively, one can assume the Gaussian prior

$$P(y_{\leq p}|\theta) = \frac{1}{(2\pi\sigma^2)^{\frac{p}{2}}} \exp \left[-y_{\leq p}^T \frac{\Gamma^{-1}}{2} y_{\leq p} \right] \quad (73.164)$$

where

$$\Gamma_{i,j} = \left\langle \underline{y}_i, \underline{y}_j \right\rangle = \gamma_{j-i} \quad (73.165)$$

for $i, j \in \mathbb{Z}_{[1,p]}$. We are assuming that all \underline{y}_τ have been demeaned; i.e., the mean

$$\hat{\mu} = \frac{1}{t} \sum_{\tau=1}^t y_\tau \quad (73.166)$$

has been subtracted from each y_τ for $\tau \in \mathbb{Z}_{[1,t]}$.

73.15.2 PL of $MA(q)$

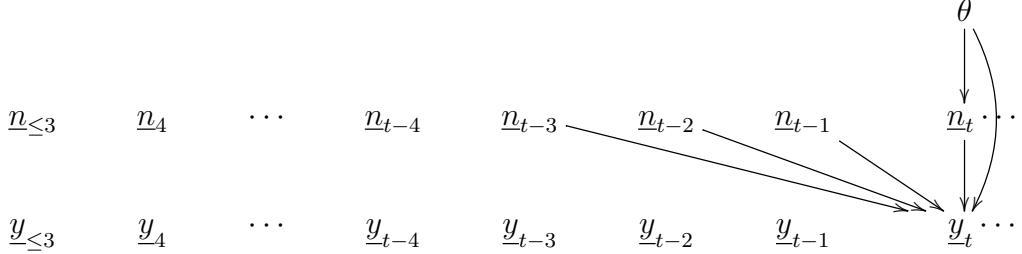


Figure 73.16: Bnet for PL of $MA(3)$. For clarity, we show only the arrows entering nodes \underline{y}_t and n_t .

Recall that if $\{y_t\}_{\forall t}$ is an $MA(q)$ t-series,

$$y_t = \sum_{j=1}^q \nu_j n_{t-j} + n_t \quad (73.167)$$

Let $\theta = (\sigma^2, \nu_{[1,q]})$. We want to learn the parameters θ of the $MA(q)$ bnet Fig.73.16, assuming the TPMs, printed in blue, are as follows.

For $t \geq q + 1$, let

$$P(y_t | n_{\leq t}, \theta) = \mathbb{1}(y_t = \text{see Eq.(73.167)}) \quad (73.168)$$

$$P(n_t | \theta) = \frac{1}{\sigma \sqrt{2\pi}} \exp \left[\frac{-(n_t)^2}{2\sigma^2} \right] \quad (73.169)$$

Note that

$$P(y_{\leq t} | \theta) = \sum_{n_{[q+1,t]}} \left\{ \prod_{\tau \in [q+1,t]} P(y_\tau | n_{[\tau-q,\tau-1]}, n_\tau) P(n_\tau | \theta) \right\} P(n_{\leq q} | \theta) \quad (73.170)$$

$$= \prod_{\tau \in [q+1,t]} \left\{ \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left[\frac{-\left(y_\tau - \sum_{j=1}^q \nu_j n_{\tau-j} \right)^2}{2\sigma^2} \right] \right\} P(n_{\leq q} | \theta) \quad (73.171)$$

The log likelihood function $\mathcal{L}_t(\theta)$ for this bnet is defined by

$$\mathcal{L}_t(\theta) = \frac{1}{t} \ln P(y_{\leq t} | \theta) \quad (73.172)$$

$$\mathcal{L}_t(\theta) = \underbrace{\frac{1}{t} \ln P(y_{\leq q} | \theta)}_{\mathcal{L}_t^{prior}(\theta)} + \mathcal{L}'_t(\theta) \quad (73.173a)$$

where

$$\mathcal{L}'_t(\theta) = \frac{-(t-q)}{t} \ln \sqrt{2\pi\sigma^2} - \frac{1}{2t\sigma^2} \sum_{\tau=q+1}^t \left(y_\tau - \sum_{j=1}^q \nu_j n_{\tau-j} \right)^2 \quad (73.173b)$$

Henceforth, we will assume that $n_{\leq q} = 0$. By Eq.(73.167), this implies that $y_{\leq q} = 0$, so the prior log-likelihood $\mathcal{L}_t^{prior}(\theta)$ is independent of θ .

Eq.(73.173) expresses $\mathcal{L}'_t(\theta)$ in terms of y_τ 's and n_τ 's, but the input data consist of only y_τ 's. Luckily, Eq.(73.167) can be used to express all the $n_{\leq t}$ in terms of the $y_{\leq t}$, assuming the input $n_{\leq q} = 0$. Next, we will explain how to do this. For definiteness and simplicity, we will assume $q = 2$ and $\nu_1 = \nu_2 = 1$, and we will use Fig.73.17 as a pedagogical aid.

For $q = 2$, Eq.(73.167) gives y_τ as a linear combination of 3 \underline{n} nodes, $\underline{n}_\tau, \underline{n}_{\tau-1}$ and $\underline{n}_{\tau-2}$. But since it's a linear equation, it can be used to express the latest \underline{n} node, i.e., \underline{n}_τ , in terms of y_τ and the 2 previous \underline{n} nodes.

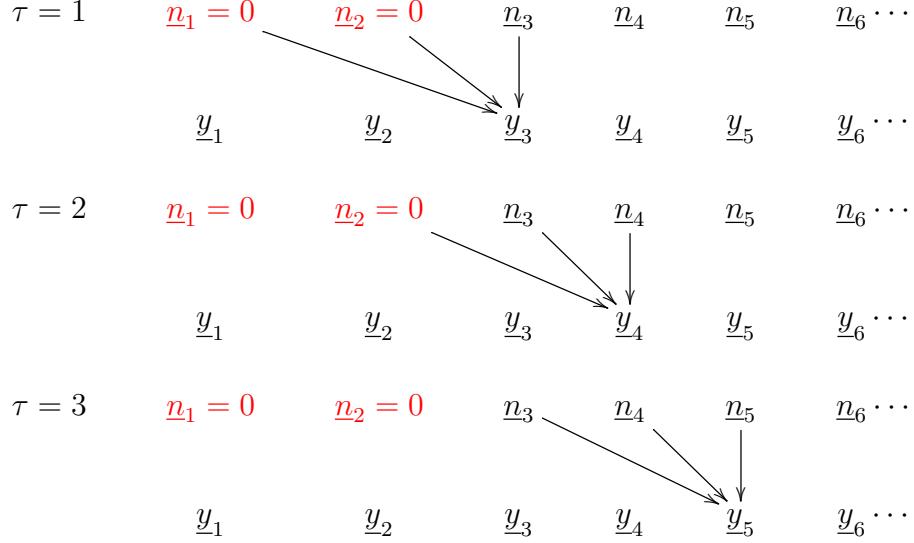


Figure 73.17: Bnet for PL of $MA(2)$. We assume $\underline{n}_1 = \underline{n}_2 = 0$. For clarity, we show only the arrows entering node $\underline{y}_{2+\tau}$ for $\tau = 1, 2, 3$.

Hence, we see from Fig.73.17 that:

From $\underline{n}_1 = \underline{n}_2 = 0$, we get $\underline{y}_1 = \underline{y}_2 = 0$.

From the $\tau = 1$ bnet, we get

$$\underline{n}_3 = \underline{y}_3 \quad (73.174)$$

From the $\tau = 2$ bnet, we get

$$\underline{n}_4 = \underline{y}_4 - \underline{n}_3 \quad (73.175)$$

From the $\tau = 3$ bnet, we get

$$\underline{n}_5 = \underline{y}_5 - \underline{n}_3 - \underline{n}_4 \quad (73.176)$$

73.15.3 PL of $ARMA(p, q)$

Recall that if $\{\underline{y}_t\}_{\forall t}$ is an $AR(p, q)$ t-series, then

$$y_t = \sum_{j=1}^p \alpha_j y_{t-j} + n_t + \sum_{j=1}^q \nu_j y_{t-j} \quad (73.177)$$

Let $\theta = (\sigma^2, \alpha_{[1,p]}, \nu_{[1,q]})$. We want to learn the parameters θ of the $ARMA(p, q)$ bnet Fig.73.18, in which the TPMs, printed in blue, are as follows.

For $t \geq \max(p+1, q+1)$, let

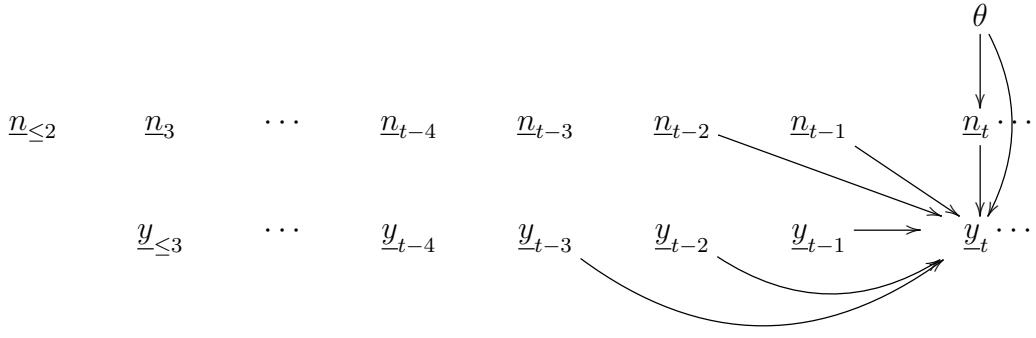


Figure 73.18: Bnet for PL of $AR(3, 2)$. For clarity, we show only the arrows entering nodes y_t and n_t .

$$P(y_t | y_{<t}, n_{\leq t}, \theta) = \mathbb{1} (y_t = \text{see Eq.(73.177)}) \quad (73.178)$$

$$P(n_t | \theta) = \frac{1}{\sigma \sqrt{2\pi}} \exp \left[\frac{-(n_t)^2}{2\sigma^2} \right] \quad (73.179)$$

As we did for $AR(p)$ and $MA(q)$, we can now calculate a log likelihood function and maximize it to obtain the parameters θ . But first we will have to assume that $n_{\leq q} = 0$ and $y_{[1,t]}$ is known, and we will have to express the unknown $n_{[q+1,t]}$ in terms of the known $y_{[1,t]}$ using the t-series definition Eq.(73.177).

73.16 VAR(p)

Let $\vec{x}_t = (\underline{x}_t^1, \underline{x}_t^2, \dots, \underline{x}_t^{nr})^T \in \mathbb{R}^{nr \times 1}$. Thus, \vec{x}_t for each t is a column vector with nr rows. A **vector time series**, denoted variously by $\{\vec{x}_1, \vec{x}_2, \dots, \vec{x}_{nt}\} = \{\vec{x}_t\}_{t=1}^{nt} = \{\vec{x}_t\}_{\forall t}$, is a set of $\mathbb{R}^{nr \times 1}$ vectors index by a discrete set of times $\mathbb{Z}_{[0,nt]}$.

Henceforth, we will use the Einstein summation convention for uppercase indices like $A, B, C \in \mathbb{Z}_{[1,nr]}$; i.e., they should be summed over from 1 to nr if they are repeated in a product. For example, $x^A y^A = \sum_{A=1}^{nr} x^A y^A$.

By **vector white noise** $\{\vec{n}_t\}_{\forall t} \sim WN(0, \Sigma)$, where $\Sigma \in \mathbb{R}^{nr \times nr}$, we mean a vector t-series $\{\vec{n}_t\}_{\forall t}$ that satisfies, for $A, B \in \mathbb{Z}_{[1,nr]}$ and $t, t' \in \mathbb{Z}_{[1,nt]}$,

$$E[n_t^A] = 0 \quad (73.180)$$

$$\langle \underline{n}_t^A, \underline{n}_{t'}^B \rangle = \Sigma^{A,B} \delta(t, t') \quad (73.181)$$

Suppose $\{\vec{y}_t\}_{\forall t}$ is a zero mean vector t-series. Hence $\underline{y}_t^A = \underline{Y}_t^A - \mu^A$, $E[\underline{Y}_t^A] = \mu^A$, $E[\underline{y}_t^A] = 0$. Suppose also that $\{\vec{n}_t\}_{\forall t} \sim WN(0, \Sigma)$. Then we define a **Vector Auto-Regressive t-series VAR(p)** by

$$\underline{y}_t^A = \sum_{j=1}^p \alpha_j^{A,B} \underline{y}_{t-j}^B + \underline{n}_t^A \quad (73.182)$$

The bnet for $VAR(p)$ is the same as the bnet for $AR(p)$ except that now, for all t , the nodes \underline{y}_t and \underline{n}_t are replaced by \vec{y}_t and \vec{n}_t and now the node weights α_t are $nr \times nr$ matrices with entries $\alpha_t^{A,B}$. For example, for $nr = 2$, we have

$$\begin{bmatrix} \underline{y}_t^1 \\ \underline{y}_t^2 \end{bmatrix} = \sum_{j=1}^p \begin{bmatrix} \alpha_j^{1,1} & \alpha_j^{1,2} \\ \alpha_j^{2,1} & \alpha_j^{2,2} \end{bmatrix} \begin{bmatrix} \underline{y}_{t-j}^1 \\ \underline{y}_{t-j}^2 \end{bmatrix} + \begin{bmatrix} \underline{n}_t^1 \\ \underline{n}_t^2 \end{bmatrix} \quad (73.183)$$

Chapter 74

Transfer Learning

Historically, the term Transfer Learning (TL) has been used in AI mostly by Artificial Neural Net (ANN) proponents (see the Wikipedia article on TL, Ref.[130]). Recently, however, a theory of causal TL has begun to emerge (see Chapter 76).

TL in AI is a fairly wide topic that is somewhat ambiguously defined. Some subjects that can be lumped under the heading of TL in AI are:

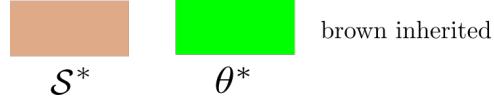
- data fusion/combining models
- model generalization
- transportability of causal knowledge, external validity (see Chapter 76)

Most AI researchers will agree that it is highly desirable to have TL in AI, because the human brain obviously does plenty of TL to great advantage. Although reams of papers have been written about the subject of TL in AI, a systematic theory of TL in AI that is universally accepted and popular remains elusive. The current theory of TL for ANN looks to me like a grab bag of heuristic approaches that are fragile, meaning they can be easily spoofed. The theory of TL for bnets (see Chapter 76) seems to me to be in much better shape: it's more elaborate, systematic, and it yields more robust results.

In this brief chapter, I will limit myself to describing a possible way of classifying, from a bnet perspective, the various approaches to TL in AI. Note that this method of classification even works for TL for ANNs, because ANNs can be viewed as bnets with deterministic nodes and a layered structure. One can describe TL in AI as a systematic way of defining a bnet \mathcal{B}^* using a bnet \mathcal{B} and other information. Bnet \mathcal{B} is associated with a dataset \mathcal{D} , and bnet \mathcal{B}^* is associated with a dataset \mathcal{D}^* . A bnet $\mathcal{B} = (\mathcal{S}, \theta)$ comprises a DAG structure \mathcal{S} and a TPM for each node of the DAG. We'll denote the TPMs (aka parameters) of \mathcal{B} by θ . So let's classify the various approaches to TL in AI by specifying what parts of the structure \mathcal{S} and parameters θ of \mathcal{B} are transferred to the structure \mathcal{S}^* and parameters θ^* of \mathcal{B}^* , and what parts of $\mathcal{B}^* = (\mathcal{S}^*, \theta^*)$ are new.

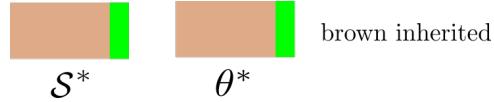
1. **Fine tune parameters.** $\mathcal{S}^* = \mathcal{S}$, $\theta^* \approx \theta$.

In this approach, we use the dataset \mathcal{D}^* associated with bnet \mathcal{B}^* to adjust slightly the parameters from θ to θ^* .

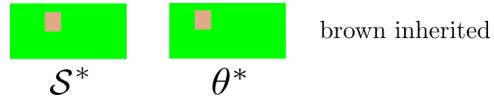


2. **Replace final layers of \mathcal{S} and of θ by new ones.** $\mathcal{S}^* = \mathcal{S}$ and $\theta^* = \theta$ except for final layers,

For example, in an ANN, replace the final layers by new ones, and use \mathcal{D}^* to find the parameters of those new final layers.



3. **Transfer only the TPM of a single node y of \mathcal{B} .** \mathcal{S}^* and θ^* are new except $P^*(y|pa(y)) = P(y|pa(y))$.



Chapter 75

Transformer Networks

This chapter is based on Refs.[18] and [131].

Transformer Networks (TN) have been taking the field of Natural Language Processing (NLP) by storm in recent years. They were introduced in 2017 and already are the basis of BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-trained Transformer), two TN libraries that have been trained with huge databases such as all of English Wikipedia (2,500M words).

Recurrent Neural Nets (RNNs) are discussed in Chapter 59. TNs are quickly displacing RNNs, an older method, in NLP. TNs are better than RNNs for doing NLP in several important ways. Whereas RNNs analyze the tokens (words) of a sentence sequentially (like a Kalman Filter), TNs analyze them in parallel, and thus are more amenable to parallel computing. Also, because RNNs analyze the words of a sentence sequentially, they tend to give more importance to the end of a sentence than to its beginning. That's because RNNs start forgetting the beginning of a sentence by the time they reach its end, like a patient with Alzheimer's. TNs do not suffer from this malady.

Dynamical bnets are discussed in Chapter 22. In Chapter 59, we showed that RNNs are dynamical bnets. The goal of this chapter is to define TNs, and to show that they too are dynamical bnets.

Let

\mathcal{S} be the set of words in a sentence,

$h_i^t \in \mathbb{R}^{nh}$ be an nh dimensional column vector for word $i \in \mathcal{S}$ at time t .

$Q^t, K^t, V^t \in \mathbb{R}^{nh \times nh}$ be the **prior-attention matrices for time slice t** .

These matrices are learned by training the net. The letters Q, K, V stand for Query, Key and Value, respectively.

Fig.75.1 represents a TN of a 3-word sentence as a dynamical bnet. The TPMs, printed in blue, for bnet Fig.75.1, are as follows:

$$P(V^t) = \delta(V^t, V_0^t) \quad (75.1a)$$

$$\begin{array}{ccc}
\underline{V}^t & \underline{Q}^t & \underline{K}^t \\
\downarrow & \downarrow & \downarrow \\
\{\underline{v}_i^t\}_{i=0,1,2} & \{\underline{q}_i^t\}_{i=0,1,2} & \{\underline{k}_i^t\}_{i=0,1,2}
\end{array}$$

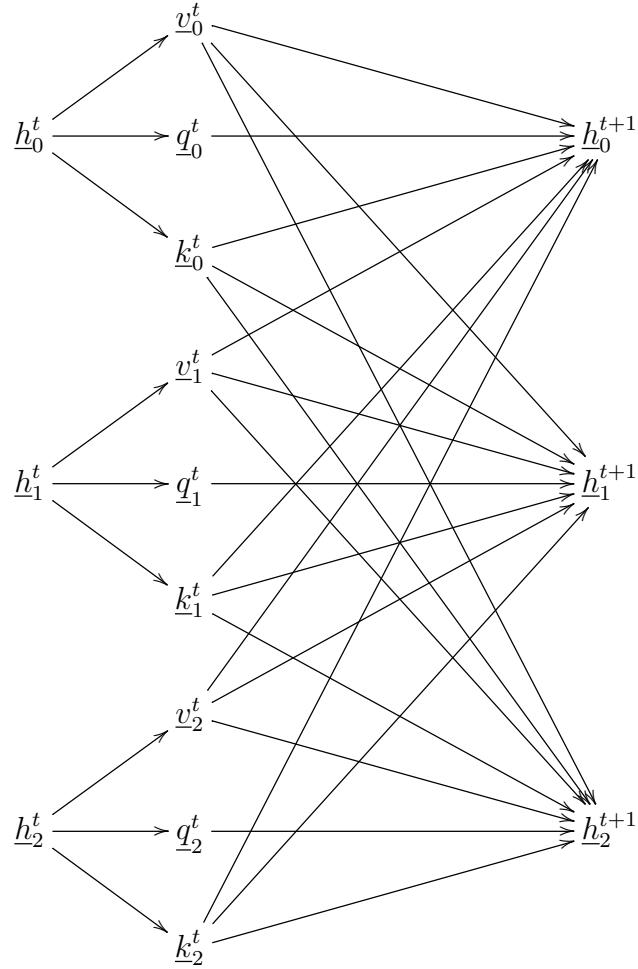


Figure 75.1: Time slice t of dynamical bnet for a transformer network (TN) of a 3-word sentence. The following arrows were not drawn explicitly for clarity: arrows pointing from node \underline{V}^t to nodes \underline{v}_i^t for $i = 0, 1, 2$, from node \underline{Q}^t to nodes \underline{q}_i^t for $i = 0, 1, 2$, and from node \underline{K}^t to nodes \underline{k}_i^t for $i = 0, 1, 2$. Note that k_i^t for all i points to h_j^{t+1} for all j . Likewise, \underline{v}_i^t for all i points to h_j^{t+1} for all j .

$$P(Q^t) = \delta(Q^t, Q_0^t) \quad (75.1b)$$

$$P(K^t) = \delta(K^t, K_0^t) \quad (75.1c)$$

$$P(v_i^t | V^t, h_i^t) = \mathbb{1}(-v^t = V^t h_i^t) \quad (75.2)$$

$$P(q_i^t | Q^t, h_i^t) = \mathbb{1}(-q^t = Q^t h_i^t) \quad (75.3)$$

$$P(k_i^t | K^t, h_i^t) = \mathbb{1}(-k^t = K^t h_i^t) \quad (75.4)$$

$$P(h_i^{t+1} | v_i^t, q_i^t, k_i^t) = \mathbb{1}\left(-h_i^{t+1} = \sum_{j \in \mathcal{S}} v_j^t \underbrace{\frac{e^{(q_i^t)^T k_j^t}}{\sum_{j' \in \mathcal{S}} e^{(q_i^t)^T k_{j'}^t}}}_{\substack{w_{j|i} = [\text{softmax}((q_i^t)^T k_j^t)]_j \\ E_{j|i}[v_j^t] = \text{Attention}}} \right) \quad (75.5)$$

In Eqs.75.1, the 3 root nodes $\underline{V}^t, \underline{Q}^t, \underline{K}^t$ have delta function priors. To improve stability of the dynamical bnet, it is more common instead to use for these priors, multiple delta functions (aka **multi-heads**) labeled $c = 0, 1, \dots, nc - 1$, as follows.

$$P(V^t | c) = \delta(V^t, V_c^t) \quad (75.6a)$$

$$P(Q^t | c) = \delta(Q^t, Q_c^t) \quad (75.6b)$$

$$P(K^t | c) = \delta(K^t, K_c^t) \quad (75.6c)$$

for $c = 0, 1, \dots, nc - 1$. Even better, one can go fully Bayesian, and make these 3 prior distributions arbitrary categorical distributions to be determined by net training, instead of mere delta functions.

Chapter 76

Transportability of Causal Knowledge

This chapter is mostly based on Refs.[41] and [29].

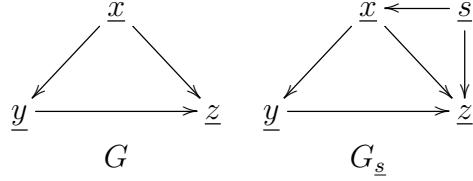


Figure 76.1: Example of selection bnet $G_{\underline{s}}$ created from bnet G .

Suppose one wants to transfer causal knowledge from a **source population** Σ to a **target population** Σ^* .

Given a bnet G , define a **selection diagram** $G_{\underline{s}}$ as a bnet formed by adding to G a new root node \underline{s} and new arrows pointing from **selection node** \underline{s} to one or more **target nodes** of G . We'll call the set of target nodes of \underline{s} the **target set** $T_{\underline{s}}$. $\underline{s} = 0$ corresponds to population Σ and $\underline{s} = 1$ to population Σ^* . For bnet G , the TPM for a node \underline{x} with parents $pa(\underline{x})$, is given by:

$$P_G(\underline{x}|pa(\underline{x})) = P(\underline{x}|pa(\underline{x})) \quad (76.1)$$

For bnet $G_{\underline{s}}$, nodes \underline{x} with parents $pa(\underline{x})$, where $\underline{s} \notin pa(\underline{x})$, have TPMs:

$$P_{G_{\underline{s}}}(\underline{x}|pa(\underline{x})) = P(\underline{x}|pa(\underline{x})). \quad (76.2)$$

Nodes \underline{x} with parents $pa(\underline{x}) \cup \underline{s}$, have TPMs:

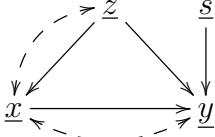
$$P_{G_{\underline{s}}}(\underline{x}|pa(\underline{x}), \underline{s}) = \begin{cases} P(\underline{x}|pa(\underline{x}), \underline{s} = 0) = P(\underline{x}|pa(\underline{x})) & \text{if } \underline{s} = 0 \\ P(\underline{x}|pa(\underline{x}), \underline{s} = 1) = P^*(\underline{x}|pa(\underline{x})) & \text{if } \underline{s} = 1 \end{cases} \quad (76.3)$$

Fig.76.1 shows an example of a selection diagram $G_{\underline{s}}$. In that figure, the target set of \underline{s} is $\{\underline{x}, \underline{z}\}$.

All this can be generalized so as to have more than one selection node, with the target sets of the selection nodes being disjoint, and such that a selection node can have more than 2 states.

Do-transport formulae

Claim 107 (*Trivial Memoryless Transportability, from Ref.[41]*)

If  where $\underline{s} \in \{0, 1\}$ is a selection node, then

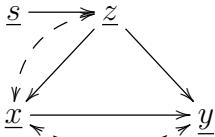
$$P^*(y|\mathcal{D}\underline{x} = x, z) = P^*(y|x, z) \quad (\text{replace } \mathcal{D} \text{ by } 1, \text{ keep } P^*) \quad (76.4)$$

$$\begin{array}{ccc} z & \xrightarrow{\underline{s}=1} & y \\ \swarrow & & \downarrow \\ \mathcal{D}\underline{x} = x & \xrightarrow{\hspace{1cm}} & y \end{array} = \begin{array}{ccc} z & \xrightarrow{\underline{s}=1} & y \\ \swarrow & & \downarrow \\ x & \xrightarrow{\hspace{1cm}} & y \end{array} \quad (76.5)$$

proof: See Claim 25.

QED

Claim 108 (*Direct Transportability, a.k.a. External Validity, from Ref.[41]*)

If  where $\underline{s} \in \{0, 1\}$ is a selection node, then

$$P^*(y|\mathcal{D}\underline{x} = x, z) = P(y|\mathcal{D}\underline{x} = x, z) \quad (\text{replace } P^* \text{ by } P, \text{ keep } \mathcal{D}) \quad (76.6)$$

$$\begin{array}{ccc} \underline{s}=1 & \longrightarrow & z \\ & \searrow & \\ \mathcal{D}\underline{x} = x & \xrightarrow{\hspace{1cm}} & y \end{array} = \begin{array}{ccc} z & \searrow & \\ \mathcal{D}\underline{x} = x & \xrightarrow{\hspace{1cm}} & y \end{array} \quad (76.7)$$

Furthermore,

$$P^*(y|\mathcal{D}\underline{x} = x) = \sum_z P(y|\mathcal{D}\underline{x} = x, z)P^*(z) \quad (76.8)$$

$$\begin{array}{ccc} \underline{s}=1 & & \underline{s}=1 \longrightarrow \sum z \\ & \searrow & \swarrow \\ \mathcal{D}\underline{x} = x & \xrightarrow{\hspace{1cm}} & y \end{array} = \begin{array}{ccc} \underline{s}=1 \longrightarrow \sum z & & \\ & \searrow & \\ \mathcal{D}\underline{x} = x & \xrightarrow{\hspace{1cm}} & y \end{array} \quad (76.9)$$

proof: See Claim 26.

QED

Claim 109 (*S-Admissible Transportability, from Ref.[41]*)

If $\underline{s} \xrightarrow{\text{---}} \underline{z} \xrightarrow{\text{---}} \underline{a}$ where $\underline{s} \in \{0, 1\}$ is a selection node, then

```

graph TD
    s[s] -- " " --> z[z]
    z -- " " --> a[a]
    s -.-> a
  
```

$$P^*(y|\mathcal{D}\underline{x} = x) = \sum_a P(y|\mathcal{D}\underline{x} = x, a)P^*(a) \quad (76.10)$$

$$\begin{array}{ccc} \underline{s} = 1 & & \underline{s} = 1 \longrightarrow \sum a \\ \searrow & & \downarrow \\ \mathcal{D}\underline{x} = x \longrightarrow y & = & \mathcal{D}\underline{x} = x \longrightarrow y \end{array} \quad (76.11)$$

proof: See Claim 27.

QED

Claim 110 (*Non-transportability, from Ref.[41]*)

If $\underline{s} \xrightarrow{\text{---}} \underline{h} \xrightarrow{\text{---}} \underline{z}$ where $\underline{s} \in \{0, 1\}$ is a selection node, then

```

graph TD
    s[s] -- " " --> h[h]
    h -- " " --> z[z]
    s -.-> z
  
```

$$P^*(y|\mathcal{D}\underline{x} = x) = P^*(y|\mathcal{D}\underline{x} = x) \quad (76.12)$$

$$\begin{array}{ccc} \underline{s} = 1 & & \underline{s} = 1 \\ \downarrow & & \downarrow \\ \mathcal{D}\underline{x} = x \longrightarrow y & = same & \end{array} \quad (76.13)$$

proof: See Claim 28.

QED

Claim 111 (*from Ref.[41]*)

If $\underline{s} \xrightarrow{\text{---}} \underline{h} \xrightarrow{\text{---}} \underline{z}$ where $\underline{s} \in \{0, 1\}$ is a selection node, then

```

graph TD
    s[s] -- " " --> h[h]
    h -- " " --> z[z]
    s -.-> z
    h ..> z
  
```

$$P^*(y|\mathcal{D}\underline{x} = x) = P(y|\mathcal{D}\underline{x} = x) \quad (76.14)$$

$$\begin{array}{ccc}
 \underline{s} = 1 & & (76.15) \\
 \searrow & & \\
 \mathcal{D}\underline{x} = x \longrightarrow y & = & \mathcal{D}\underline{x} = x \longrightarrow y
 \end{array}$$

proof: See Claim 29.

QED

Claim 112 (from Ref.[41])

If \underline{s} is a selection node, then

```

    graph TD
      s((s)) --> h((h))
      h --> z((z))
      z --> y((y))
      s -.-> z
      s -.-> y
  
```

$$P^*(y|\mathcal{D}\underline{x} = x) = \sum_z P(y|\mathcal{D}\underline{x} = x, z)P^*(z|x) \quad (76.16)$$

$$\begin{array}{ccccc}
 \underline{s} = 1 & \mathcal{D}\underline{x} = x & \underline{s} = 1 & \mathcal{D}\underline{x} = x & (76.17) \\
 \searrow & \downarrow & \searrow & \downarrow & \\
 y & = & x \longrightarrow \sum z \longrightarrow y & &
 \end{array}$$

proof: See Claim 30.

QED

Chapter 77

Turbo Codes

This chapter is based on Ref.[27].

In this chapter, vectors with n components will be indicated by an n superscript. For example, $a^n = (a_0, a_1, \dots, a_{n-1})$.

Consider an n -letter message $u^n = (u_0, u_1, \dots, u_{n-1})$, where for all i , $u_i \in \mathcal{A}$ is an element of an alphabet \mathcal{A} , and where for all i , the u_i are i.i.d.. Suppose u^n is encoded deterministically in two different ways, $e_1(u^n)$ and $e_2(u^n)$. After passing through the same memoryless channel, the variables u^n, e_1, e_2 become $\tilde{u}^n, \tilde{e}_1, \tilde{e}_2$, respectively. The letter u stands for unencoded, and e for encoded. Quantities with a tilde $\tilde{u}^n, \tilde{e}_1, \tilde{e}_2$ occur after channel passage and are visible (measurable). Quantities without a tilde u^n, e_1, e_2 are hidden (unmeasurable).

The situation just described can be represented by the bnet Fig.77.1, or by its abridged version Fig.77.2. But note that the abridged version does not show explicitly that the u_i are i.i.d. or that the channel is memoryless (i.e., that the u_i for all i pass independently through the channel).

Define

$$x = (u^n, e_1, e_2) \quad (77.1)$$

and

$$\tilde{x} = (\tilde{u}^n, \tilde{e}_1, \tilde{e}_2) . \quad (77.2)$$

Fig.77.1 implies that

$$P(x, \tilde{x}) = P(\tilde{u}^n | u^n) \left[\prod_{r=1,2} P(\tilde{e}_r | e_r) P(e_r | u^n) \right] P(u^n) . \quad (77.3)$$

Because the u^n are i.i.d.,

$$P(u^n) = \prod_i P(u_i) . \quad (77.4)$$

Because the channel is memoryless,

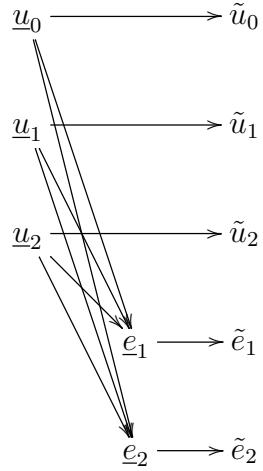


Figure 77.1: Turbo coding Bnet representing a message being encoded two different ways and then the original message and the 2 encodings pass through a memoryless channel.

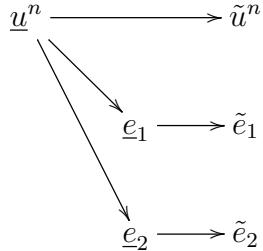


Figure 77.2: Abridged version of Fig.77.1.

$$P(\tilde{u}^n | u^n) = \prod_i P(\tilde{u}_i | u_i) . \quad (77.5)$$

Because the encoding is deterministic, we must have for $r = 1, 2$

$$P(e_r | u^n) = \delta(e_r, e_r(u^n)) . \quad (77.6)$$

Define the belief functions

$$BEL_i = BEL_i(\underline{u}_i = a) = P(\underline{u}_i = a | \tilde{x}) . \quad (77.7)$$

The best estimate of u_j given all visible evidence \tilde{x} is

$$\hat{u}_i = \operatorname{argmax}_{u_i} BEL_i(u_i) . \quad (77.8)$$

Define the probability functions

$$\pi_i = \pi_i(u_i) = P(u_i) , \quad (77.9)$$

and the likelihood functions

$$\lambda_i = \lambda_i(u_i) = P(\tilde{u}_i|u_i) . \quad (77.10)$$

For $r = 1, 2$, define the Kernel functions

$$K_r = K_r(u^n) = P(\tilde{e}_r|e_r = e_r(u^n)) . \quad (77.11)$$

In this book, $\mathcal{N}(!a)$ denotes a normalization constant that does not depend on a . Define

$$\mathcal{N}_i = \mathcal{N}(!u_i) . \quad (77.12)$$

Claim 113

$$BEL_i = \mathcal{N}_i \lambda_i \pi_i \mathcal{T}_i^{K_1 K_2} [\prod_{j \neq i} \lambda_j \pi_j] , \quad (77.13)$$

where $\mathcal{T}_i^K(\cdot)$ with $K = K_1 K_2$ is an operator (transform) that acts on functions of u^n :

$$\mathcal{T}_i^K(\cdot) = \sum_{u^n} \delta(u_i, a) K(u^n)(\cdot) . \quad (77.14)$$

proof:

$$\begin{aligned} P(\underline{u}_i = a | \tilde{x}) &= \\ &= \sum_x \delta(u_i, a) P(x | \tilde{x}) \end{aligned} \quad (77.15)$$

$$= \sum_x \delta(u_i, a) \frac{P(\tilde{x}|x)P(x)}{P(\tilde{x})} \quad (77.16)$$

$$= \mathcal{N}(!a) \sum_x \delta(u_i, a) P(\tilde{x}|x)P(x) \quad (77.17)$$

$$= \mathcal{N}(!a) \sum_x \delta(u_i, a) P(u^n) \left[\prod_{r=1,2} P(\tilde{e}_r|e_r) \delta(e_r, e_r(u^n)) \right] \prod_j P(\tilde{u}_j|u_j) \quad (77.18)$$

$$= \mathcal{N}(!a) \lambda_i(a) \pi_i(a) R , \quad (77.19)$$

where

$$R = \sum_{u^n} \delta(u_i, a) \left[\prod_{r=1,2} P(\tilde{e}_r | e_r(u^n)) \right] \prod_{j \neq i} P(\tilde{u}_j | u_j) P(u_j) \quad (77.20)$$

$$= \sum_{u^n} \delta(u_i, a) \left[\prod_{r=1,2} K_r(u^n) \right] \prod_{j \neq i} \lambda_j(u_j) \pi_j(u_j) \quad (77.21)$$

$$= \mathcal{T}_i^{K_1 K_2} \left[\prod_{j \neq i} \lambda_j(u_j) \pi_j(u_j) \right]. \quad (77.22)$$

Hence

$$BEL_i(a) = \mathcal{N}(!a) \lambda_i(a) \pi_i(a) \mathcal{T}_i^{K_1 K_2} \left[\prod_{j \neq i} \lambda_j(u_j) \pi_j(u_j) \right]. \quad (77.23)$$

QED

77.1 Decoding Algorithm

The Turbo algorithm for decoding the encode message is as follows. For $m = 0$, let

$$\pi_j^{(0)}(u_j) = \frac{1}{n_{\underline{u}_j}}. \quad (77.24)$$

Then for $m = 1, 2, \dots$, let

$$\pi_i^{(m)} = \mathcal{N}_i \mathcal{T}_i^{K_m \% 2} \left[\prod_{j \neq i} \lambda_j \pi_j^{(m-1)} \right], \quad (77.25)$$

where $m \% 2 = 1$ if m is odd and $m \% 2 = 2$ if m is even. Furthermore, for $m > 0$, let

$$BEL_i^{(m)} = \mathcal{N}_i \lambda_i \pi_i^{(m-1)} \pi_i^{(m)} \quad (77.26)$$

$$= \mathcal{N}_i \lambda_i \pi_i^{(m-1)} \mathcal{T}_i^{K_m \% 2} \left[\prod_{j \neq i} \lambda_j \pi_j^{(m-1)} \right]. \quad (77.27)$$

As $m \rightarrow \infty$, $BEL_i^{(m)}$ given by Eq.(77.27) is expected to converge to the the exact BEL_i given by Eq.(77.13).

Turbo decoding can be represented by the bnets Figs.77.3 and 77.4.

The TPMs, printed in blue, for bnet Fig.77.3, are as follows.

$$P(d_i^{(m)} = a | \tilde{u}^n, \tilde{e}_{m \% 2}) = BEL_i^{(m)}(a). \quad (77.28)$$

The TPMs, printed in blue, for bnet Fig.77.4, are as follows.

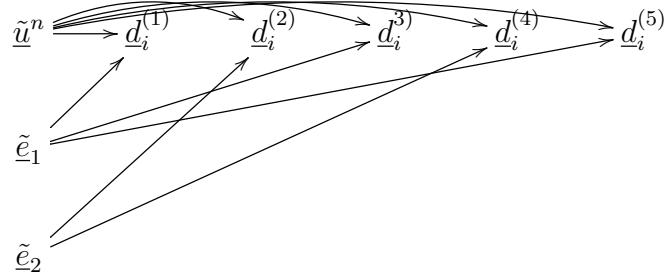


Figure 77.3: Bnet describing Turbo code generation of $BEL_i^{(m)}(a)$ for $m = 1, 2, \dots$

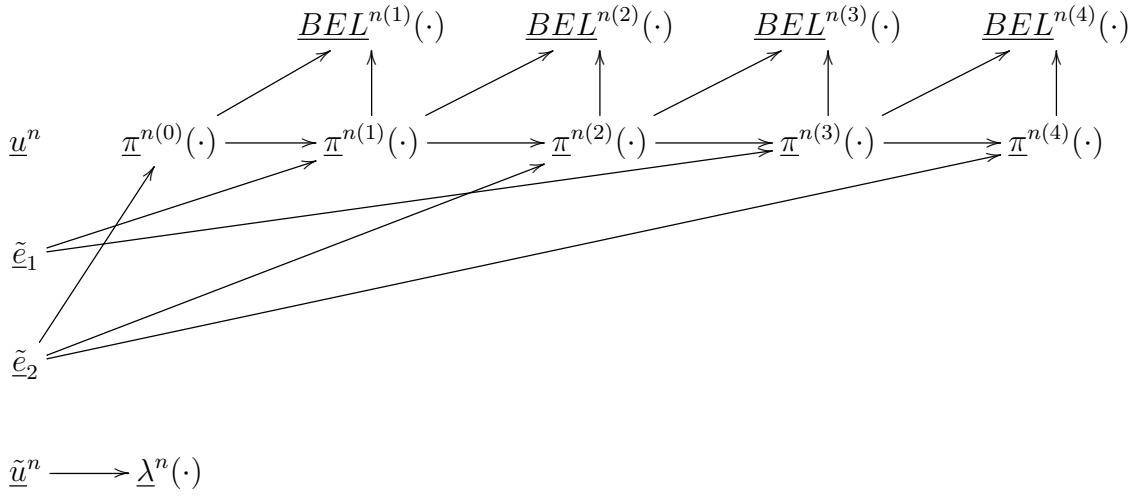


Figure 77.4: Bnet describing Turbo code generation of $BEL^{n(m)}(\cdot)$ and $\pi^{n(m)}(\cdot)$ for $m = 0, 1, 2, \dots$. The following arrows were not drawn for clarity: Arrows pointing from node $\underline{\lambda}^n(\cdot)$ to nodes $\underline{\pi}^{n(m)}(\cdot)$ and $\underline{BEL}^{n(m)}(\cdot)$ for $m = 0, 1, 2, \dots$

$$P((\lambda^n)'(\cdot) | \tilde{u}^n) = \delta((\lambda^n)'(\cdot), \lambda^n(\cdot)) \quad (77.29)$$

$$P(\pi^{n(m)}(\cdot) | \lambda^n(\cdot), \pi^{n(m-1)}(\cdot), \tilde{e}_{m \% 2}) = \prod_i \prod_{u_i} \delta(\pi_i^{(m)}(u_i), \mathcal{N}_i \mathcal{T}_i^{K_{m \% 2}} [\prod_{j \neq i} \lambda_j \pi_j^{(m-1)}]) \quad (77.30)$$

$$P(BEL^{n(m)}(\cdot) | \lambda^n(\cdot), \pi^{n(m)}(\cdot), \pi^{n(m-1)}(\cdot)) = \prod_i \prod_{u_i} \delta(BEL_i(u_i), \mathcal{N}_i \lambda_i \pi_i^{(m-1)} \pi_i^{(m)}) \quad (77.31)$$

77.2 Message Passing Interpretation of Decoding Algorithm

Ref.[27] shows that the Turbo code decoding algo can be interpreted as an application of Message Passing. We leave all talk of Message Passing to a separate chapter, Chapter 43.

Chapter 78

Uplift Modelling

This chapter is based on many references, including Ref.[13, 9, 132, 47].

Uphill Modelling (UP) deals with the application of Rubin’s Theory of Potential Outcomes (PO) to advertisement and marketing.

PO, which is discussed in Chapter 56, is a subset of Pearl’s Causal Inference. Besides UP, other applications of PO theory that are discussed in this book are: Regression Discontinuity (Chapter 60), Difference-in-Differences (Chapter 15) and Synthetic Controls (Chapter 72).

In UP, each **participant person** is interrogated at two well anticipated, fairly closely spaced times t_0 and t_1 (as opposed to Difference-in-Differences (DID), where t_0 and t_1 might be years apart, and long before the DID analysis is attempted.). In between those two times, a treatment which we will refer to as the **UP diagnostic test** is applied. For example, at times t_0 and t_1 , every participant might be asked how important he/she rates climate change on a scale of 1 to 10. In between times t_0 and t_1 , every participant might be sent a brochure on climate change. In UP, as in all other PO applications, each **sample** σ is in the treated or control groups, but not both. But in UP, the same participant can be in both the treated and control groups. If so, that participant is considered two different samples σ ; for example, $\sigma = \text{treatedBob}, \text{controlBob}$. In UP, the samples are aware of which of those groups they are in, so they are not “treatment blind”.

78.1 UP types

Let $y_t^B \in \mathbb{R}$ for $t = t_0, t_1$ be the treatment response at time t for participant B . (We are using here the same notation as in Chapter 56). Call $\delta^B = y_{t_1}^B - y_{t_0}^B$ the **participant uplift** for participant B . As shown in Fig.78.1, UP classifies participants into 4 **UP-types**: Persuadables, SureThings, LostCauses, and SleepyDogs. The UP-type of a participant depends on the changes that are induced on that participant by an **UP-diagnostic-test**.

- For a **Persuadable** participant, $\delta^B > 0$.

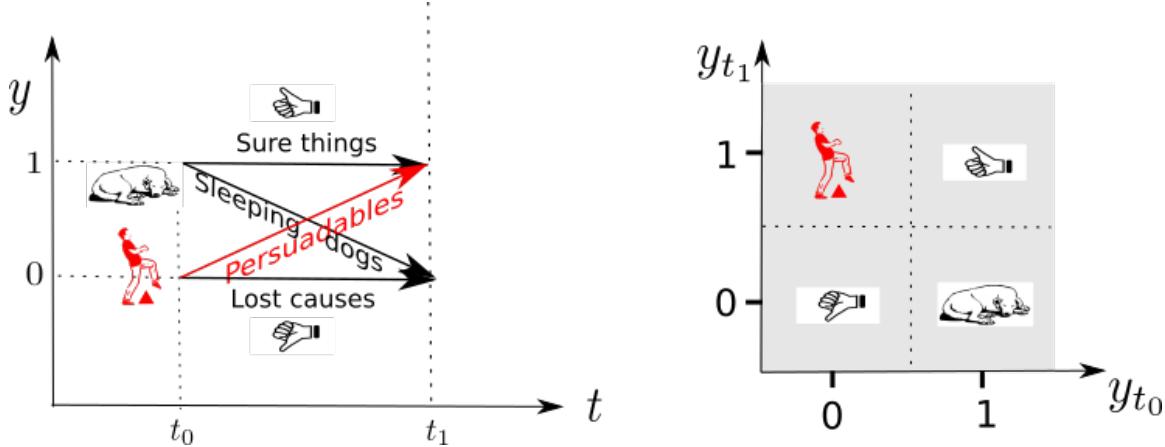


Figure 78.1: UP diagnostic test can be used to classify all participants of the population into 4 UP-types. This figure assumes $y \in \{0, 1\}$. More generally, $y \in \mathbb{R}$. t represents time. $t = t_0$ corresponds to $d = 0 = \text{untreated}$, and $t = t_1$ corresponds to $d = 1 = \text{treated}$.

- For a **SleepyDogs** participant, $\delta^B < 0$.
- For a **SureThings** participant, $\delta^B \approx 0$ and $y_{t_0}^B$ is high.
- For a **LostCauses** participant, $\delta^B \approx 0$ and $y_{t_0}^B$ is low.

Suppose B belongs to stratum A_x . What is commonly called the **uplift** is the **stratum-uplift** $\delta_x = ACE_x$. Strata can also be classified into the 4 UP-types, depending on the sign and size of their δ_x . A participant may not be typical for his stratum and may have different **participant and stratum UP-types**. For example, he may have positive participant uplift and therefore have a Persuadable participant UP-type, but his stratum-uplift might be negative, so he has the SleepyDogs stratum UP-type.

Advertisers are very interested in finding the Persuadable strata in a population so as to focus their resources on them. For example, UP was used very successfully during the Obama presidential campaigns. Team Obama conducted UP-diagnostic tests much like the climate change one described earlier. This allowed them to identify voters who might be sitting on the fence on whether to vote for Obama or not. Then Team Obama spent the lion share of resources on those fence-sitters.

78.2 Some Relevant Technical Formulas from Chapter 56

Recall the following technical formulae that were proven in Chapter 56:

- Recall Eq.(56.143):

$$ACE = \sum_x P(x) \underbrace{\sum_y y [P(y|d=1, x) - P(y|d=0, x)]}_{ACE_x} \quad (78.1)$$

If $y \in \{0, 1\}$, then

$$\underbrace{ACE_x}_{\delta_x} = \underbrace{\frac{P_{y|\underline{d}, \underline{x}}(1|1, x)}{Y_x^1}} - \underbrace{\frac{P_{y|\underline{d}, \underline{x}}(1|0, x)}{Y_x^0}} . \quad (78.2)$$

- Recall Eq.(56.158):

$$\widehat{ACE}_x = \underbrace{\frac{1}{N_x} \sum_{\sigma \in A_x} \frac{d^\sigma y^\sigma}{g_{1|x^\sigma}}}_{Y_x^1} - \underbrace{\frac{1}{N_x} \sum_{\sigma \in A_x} \frac{(1-d^\sigma)y^\sigma}{g_{0|x^\sigma}}}_{Y_x^0} \quad (78.3)$$

78.3 UP Analysis

The input to UP is a PO dataset $DS = \{(\sigma, d^\sigma, x^\sigma, y^\sigma) : \sigma = 0, 1, 2, \dots, nsam - 1\}$. where $d^\sigma \in \{0, 1\}$, $x^\sigma \in S_{\underline{x}}$, $y^\sigma \in \mathbb{R}$. A participant B is assigned two different σ if he/she belongs to both the treated and control groups. We will assume $S_{\underline{x}}$ is a finite set. In general, $x = (x_0, x_1, \dots, x_{n-1})$ is an n dimensional vector of features x_i . If any of the x_i is a priori continuous, we will assume it has been binned into a finite number of bins.

Starting with DS , UP performs the following steps. Fig.78.2 is a pictorial representation of the quantities that are calculated during these steps.

1. Find A_x for each observed $x \in S_{\underline{x}}$. Set $A_x = \emptyset$ for unobserved $x \in S_{\underline{x}}$.
2. Calculate δ_x for each $x \in S_{\underline{x}}$. Set $\delta_x = 0$ if $A_x = \emptyset$.
3. Calculate the set

$$\{\Delta_c\}_{c=0,1,\dots,nc-1} = \{\delta_x : x \in S_{\underline{x}}\} \quad (78.4)$$

of distinct uplifts δ_x . The class labels c should be assigned so that the sequence of Δ_c is monotonic and non-increasing; i.e.,

$$\Delta_0 \geq \Delta_1 \geq \dots \geq \Delta_{nc-1} . \quad (78.5)$$

Now calculate

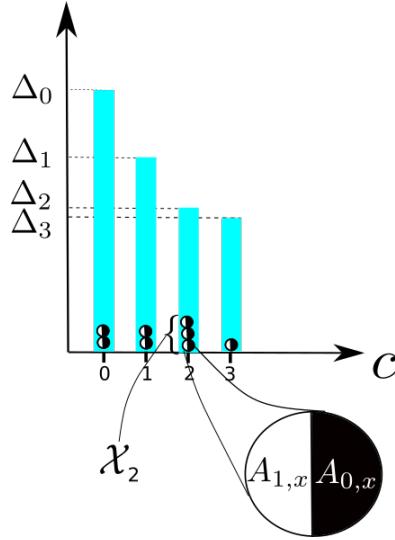


Figure 78.2: Pictorial representation of the sequence $\{(\mathcal{X}_c, \Delta_c)\}_{c=0,1,\dots,nc-1}$.

$$\mathcal{X}_c = \{x : \delta_x = \Delta_c\} \quad (78.6)$$

for each c . By the end of this step, we will have calculated $\{(\mathcal{X}_c, \Delta_c)\}_{c=0,1,\dots,nc-1}$. We will refer to the \mathcal{X}_c as **strata-bins**. Note that

$$\Delta_c = \frac{1}{|\mathcal{X}_c|} \sum_{x \in \mathcal{X}_c} \delta_x \quad (78.7)$$

$$= \underbrace{\frac{1}{|\mathcal{X}_c|} \sum_{x \in \mathcal{X}_c} Y_x^1}_{Y_c^1} - \underbrace{\frac{1}{|\mathcal{X}_c|} \sum_{x \in \mathcal{X}_c} Y_x^0}_{Y_c^0} . \quad (78.8)$$

4. For each c , calculate

$$\Sigma_{d,c} = \cup_{x \in \mathcal{X}_c} A_{d,x} \quad (78.9)$$

for $d \in \{0, 1\}$ and

$$\Sigma_c = \Sigma_{0,c} \cup \Sigma_{1,c} . \quad (78.10)$$

Fig.78.3 is a way of plotting the results of UP in an intuitive way that even a business type can understand. UP software often plots something called a Qini curve, but I find Qini curves opaque, confusingly defined in the literature, unnecessary and not very well motivated. So I don't use them.

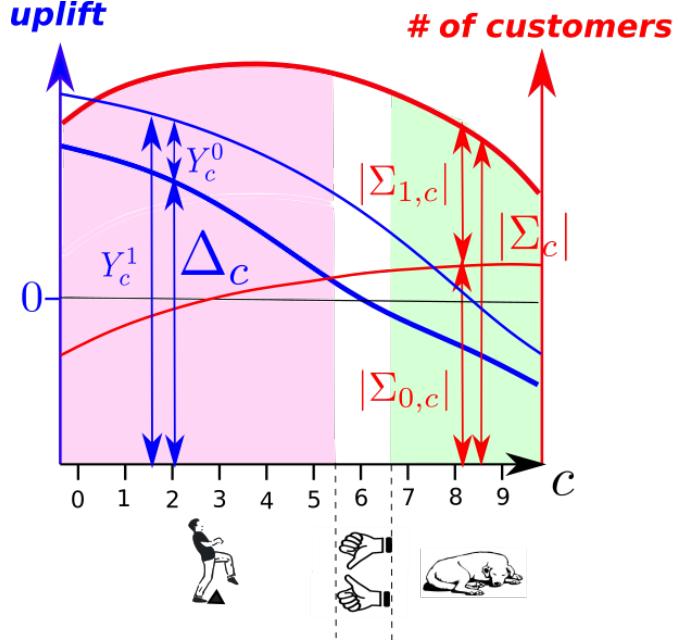


Figure 78.3: Plot of UP results. Alternative to Qini curves.

78.4 UP Decision Trees

In this section, we will describe how to build UP decision trees (UP dtrees), and explain why they are needed for UP.

Generic dtrees are described in Chapter 13. This section complements rather than replaces that chapter so the reader is advised to read that chapter first.

Ref.[47] is an excellent paper on the use of dtrees in UP.

The analysis described previously in Section 78.3, although theoretically correct, will work very poorly in practice. The strata-bins of Section 78.3 correspond to the classification classes of a dtree. But strata-bins are very specific so they severely overfit the data. Although dtrees can also suffer from overfitting, there are known methods of preventing or mitigating overfitting in dtrees.

There are also tasks that dtrees can do well and the methods explained so far cannot do well. For example, suppose we have a classless dataset $DS^- = \{(\sigma, x^\sigma) : \sigma \in \Sigma^-\}$ and we want to predict the class c^σ and uplift Δ_{c^σ} for each of these individuals $\sigma \in \Sigma^-$. A dtree can easily do that. The alternative is to use the classy dataset $DS = \{(\sigma, x^\sigma, c^\sigma) : \sigma \in \Sigma\}$ to prepare a dictionary that orders the elements of S_x and gives a class c and an uplift value Δ_c for each feature vector $x \in S_x$. But such a dictionary overfits and says nothing for feature vectors x that do not show up in the classy dataset DS ; i.e., the dictionary doesn't guess (interpolate). Dtrees, on the other hand, do guess.

So, without further ado, let us describe how to modify the results of Chapter 13 on generic dtrees to the case of UP dtrees. The main difference, as we will explain

in detail next, is that the Information Gain metric used for generic dtrees needs to be replaced by another metric.

$$\begin{aligned} & \text{Top: } \underline{x}_j \xrightarrow{\underline{x}_j = \underline{x}'_j} \underline{x}'_j \\ & \{N_j^d(c, x_j)\}_{c \in S_c, x_j \in S_{\underline{x}_j}} \\ & \sum_{c \in S_c} N_j^d(c, x_j) = N_j^d(x_j) \\ & \sum_{x_j \in S_{\underline{x}_j}} N_j^d(c, x_j) = N_j^d(c) = \sum_{x_j \in S_{\underline{x}_j}} N_j^d(x_j) = N_j^d \end{aligned}$$

Figure 78.4: Fig.13.4 with d dependence added. $d \in \{0, 1\}$ is the treatment dose.

Fig.78.4 was obtained from Fig.13.4 by adding d dependence. $d \in \{0, 1\}$ is the treatment dose. Note that in UP, we build a dtree in which every node carries a double ($d = 0, 1$) TPV. This is in contrast to the generic dtrees built in Chapter 13, in which each node carries a single TPV. $N_j^d(c, x_j)$ is the number of individuals σ in the population that reaches node \underline{x}_j with $d \in \{0, 1\}$, belonging to class $c \in S_c$ and having $\underline{x}_j = x_j$. From these population numbers, we can define the bnet in Fig.78.5. The TPMs, printed in blue, for the (non-root) nodes of this bnet, are as follows

$$P(c|x_j, j, d) = \frac{N_j^d(c, x_j)}{N_j^d(x_j)} \quad (78.11)$$

$$P(x_j|j, d) = \frac{N^d(x_j)}{N_j^d} \quad (78.12)$$

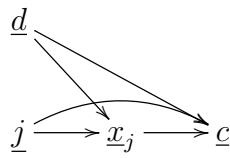


Figure 78.5: Bnet derived from population numbers in Fig.78.4

In Chapter 13, we used Information Gain (a mutual information) as the SAM (Separation Ability Measure) in SL (Structure Learning) of dtrees (Decision Trees). Information Gain is a bad SAM for SL of UP dtrees, because it knows nothing about $d = 0, 1$ and the double TPVs of nodes in UP dtrees. For UP dtrees, we need a SAM specifically designed to separate $d = 0, 1$, and generate classes that are uplift bins (i.e., uplift intervals).

Ref.[47] proposes and studies the following 3 SAMs for doing SL of UP dtrees.

1. SAM_DD (DD=Delta Delta)

For $d \in \{0, 1\}$ and $c, c' \in S_{\underline{c}}$, define the increments

$$\partial_d f(d) = f(1) - f(0) \quad (78.13)$$

and

$$\partial_{c',c} f(c) = f(c') - f(c) . \quad (78.14)$$

Let

$$\Delta_{c|j} = P(c|j, 1) - P(c|j, 0) \quad (78.15)$$

$$= \partial_d P(c|j, d) \quad (78.16)$$

$$SAM_DD_j = \max_{c,c'} |\partial_{c',c} \partial_d P(c|j, d)| \quad (78.17)$$

$$= \max_{c,c'} |\partial_{c',c} \Delta_{c|j}| \quad (78.18)$$

2. SAM_KL (KL=Kullback Liebler)

$$SAM_KL_j = \left[\sum_{x_j \in S_{\underline{x}_j}} P(x_j|j) D_{KL}(P_{\underline{c}|x_j,j,1} \| P_{\underline{c}|x_j,j,0}) \right] - D_{KL}(P_{\underline{c}|j,1} \| P_{\underline{c}|j,0}) \quad (78.19)$$

$$= \left[\sum_{x_j \in S_{\underline{x}_j}} P(x_j|j) \sum_{c \in S_{\underline{c}}} P(c|x_j, j, 1) \ln \frac{P(c|x_j, j, 1)}{P(c|x_j, j, 0)} \right] - \sum_{c \in S_{\underline{c}}} P(c|j, 1) \ln \frac{P(c|j, 1)}{P(c|j, 0)} \quad (78.20)$$

SAM_KL_j can be negative.

3. SAM_E (E=Euclidean)

SAM_E_j is defined the same way as SAM_KL_j except with the KL divergence $D_{KL}(P \parallel Q)$ in SAM_KL replaced by the Euclidean distance squared.

$$D(P, Q) = \sum_x (P(x) - Q(x))^2 \quad (78.21)$$

The intuitive reason for using these quantities as SAMs is that they maximize the change in uplift between successive tree levels, so that the uplift increases as quickly as possible as we descend down the UP tree. In the case of generic dtrees for which we use Information Gain as SAM, we are maximizing the correlation between classes and nodes as we descend down the tree. These two goals are related. In fact, in the limit where the number of control individuals becomes zero, SAM_KL_j and IG_j become the same, as will be shown later.

Next we show that SAM_KL_j satisfies the following 3 axioms¹

Claim 114 .

1. SAM_KL_j is minimum iff $P(c|x_j, j, 0) = P(c|x_j, j, 1)$ for all c and x_j .
2. If $P(c|j, d) = P(c|d)$ for all c, d , then $SAM_KL_j = 0$.
3. Suppose $N_r^0 = 0$ for all nodes $r \in J_0$ (i.e., no control population) and we use the Laplace Correction when warranted. Then

$$SAM_KL_j = H(\underline{c} : \underline{x}_j | j, 1) \quad (78.22)$$

$$= IG_j \quad \text{for treated population .} \quad (78.23)$$

proof:

The proof of items 1 and 2 follow by inspection of Eq.(78.20). Item 3 is proven in Claim 115 below.

QED

Let $N_{\underline{c}} = |S_{\underline{c}}|$. Define the uniform probability distribution

$$U_{\underline{c}}(c) = \frac{1}{N_{\underline{c}}} \quad (78.24)$$

for all $c \in S_{\underline{c}}$.

Eq.(78.11) for the TPM of node \underline{c} in the bnet Fig.78.5 can be "Laplace Corrected" as follows so that it is no longer undefined when its denominator vanishes:

¹We won't show it here, but according to Ref.[47], SAM_E_j also satisfies these 3 axioms, but SAM_DD_j satisfies only the first two.

$$P(c|j, d) = \begin{cases} \frac{N_j^d(c)}{N_j^d} & \text{if } N_j^d > 0 \\ U_{\underline{c}}(c) & \text{if } N_j^d = 0 \text{ (Laplace Correction)} \end{cases} \quad (78.25)$$

Claim 115 Suppose $N_r^0 = 0$ for all dtree nodes $r \in J_0$ and we use the Laplace Correction when warranted. Then

$$SAM_KL_j = H(\underline{c} : \underline{x}_j | j, 1) . \quad (78.26)$$

proof:

For all nodes $r \in J_0$, we must have

$$P_{\underline{c}|r,0} = U_{\underline{c}} \quad (78.27)$$

so

$$D_{KL}(P_{\underline{c}|r,1} \| P_{\underline{c}|r,0}) = D_{KL}(P_{\underline{c}|r,1} \| U_{\underline{c}}) \quad (78.28)$$

$$= \ln(N_{\underline{c}}) - H(\underline{c}|r, 1) . \quad (78.29)$$

For all $x_j \in S_{\underline{x}_j}$, we must also have

$$N_j = N_j^1, N(x_j) = N^1(x_j) \quad (78.30)$$

so

$$P(x_j | j) = P(x_j | j, 1) . \quad (78.31)$$

Now using Eqs.(78.29) and (78.31), we get

$$SAM_KL_j = - \left[\sum_{x_j \in S_{\underline{x}_j}} P(x_j | j) H(\underline{c} | x_j, j, 1) \right] + H(\underline{c} | j, 1) \quad (78.32)$$

$$= - \left[\sum_{x_j \in S_{\underline{x}_j}} P(x_j | j, 1) H(\underline{c} | x_j, j, 1) \right] + H(\underline{c} | j, 1) \quad (78.33)$$

$$= -H(\underline{c} | \underline{x}_j, j, 1) + H(\underline{c} | j, 1) \quad (78.34)$$

$$= H(\underline{c} : \underline{x}_j | j, 1) \quad (78.35)$$

QED

78.4.1 Appendix, connection between Δ_c and $\Delta_{c|j}$

Recall Eq.(78.8):

$$\begin{aligned} \Delta_c &= \underbrace{\frac{1}{|\mathcal{X}_c|} \sum_{x \in \mathcal{X}_c} Y_x^1 - \frac{1}{|\mathcal{X}_c|} \sum_{x \in \mathcal{X}_c} Y_x^0}_{Y_c^1 - Y_c^0} \quad (78.36) \\ &= \partial_d Y_c^d. \end{aligned}$$

$$(78.37)$$

Compare that to Eq.(78.16):

$$\Delta_{c|j} = P(c|j, 1) - P(c|j, 0) \quad (78.38)$$

$$= \partial_d P(c|j, d) \quad (78.39)$$

What is the connection between these 2 deltas, Δ_c and $\Delta_{c|j}$? Are they equal?

First off, notice that $\Delta_{c|j}$ is defined for all nodes j of the dtree. Let $j(c)$ be the leaf node for which $\Delta_c \approx \Delta_{c|j(c)}$. Assume $y^\sigma \in \{0, 1\}$. Then

$$P(c|j = j(c), d) = \frac{N_{j(c)}^d(c)}{N_{j(c)}^d} \approx Y_c^d \quad (78.40)$$

So the two deltas are indeed approximately equal when $y^\sigma \in \{0, 1\}$ and $j = j(c)$.

Chapter 79

Variational Bayesian Approximation

For more info and references about this topic, see Ref.[133].

The Variational Bayesian approximation (VBA) is an analytic (as opposed to numerical) approximation to the probability distribution $P(h|\vec{x})$, where h are the hidden variables and \vec{x} is the data.

More precisely, suppose $\underline{h} \in S_h$ and $\underline{q} \in S_h$. Suppose $\underline{\vec{x}} \in S_{\vec{x}}^{nsam}$ is a vector of $nsam$ samples and the samples $\underline{x}[\sigma] \in S_x$ are i.i.d.. The VBA is simply an approximation $P_{\underline{q}|\vec{x}}$ to $P_{\underline{h}|\vec{x}}$:

$$P_{\underline{h}|\vec{x}}(h|\vec{x}) \approx P_{\underline{q}|\vec{x}}(h|\vec{x}) \quad (79.1)$$

obtained by minimizing the Kullback-Liebler divergence $D_{KL}(P_{\underline{q}|\vec{x}} \| P_{\underline{h}|\vec{x}})$ over all $P_{\underline{q}|\vec{x}}$. The minimization is usually subject to some constraints on the admissible forms of $P_{\underline{q}|\vec{x}}$.

$D_{KL}(Q \| P) \neq D_{KL}(P \| Q)$; i.e., D_{KL} is not symmetric. So why do we use $D_{KL}(P_{\underline{q}|\vec{x}} \| P_{\underline{h}|\vec{x}})$ instead of $D_{KL}(P_{\underline{h}|\vec{x}} \| P_{\underline{q}|\vec{x}})$? Because $D_{KL}(P_{\underline{h}|\vec{x}} \| P_{\underline{q}|\vec{x}})$ requires knowledge of $P_{\underline{h}|\vec{x}}$, but calculating $P_{\underline{h}|\vec{x}}$ is what we are trying to do in the first place.

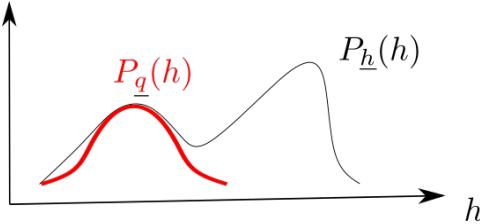


Figure 79.1: If $P_{\underline{q}}(h)$ is Gaussian shaped and $P_{\underline{h}}(h)$ has multiple bumps (modes) then $D_{KL}(P_{\underline{q}} \| P_{\underline{h}})$ is minimized when $P_{\underline{q}}$ fits one of the modes of $P_{\underline{h}}$. That is because $D_{KL}(P_{\underline{q}} \| P_{\underline{h}}) = \sum_h P_{\underline{q}}(h) \ln \frac{P_{\underline{q}}(h)}{P_{\underline{h}}(h)}$ is a weighted average with weights $P_{\underline{q}}$, so nothing going on outside the support of $P_{\underline{q}}$ influences much the final average.

See Fig.79.1 for some intuition on what minimizing $D_{KL}(P_{\underline{q}|\vec{x}} \| P_{\underline{h}|\vec{x}})$ means.

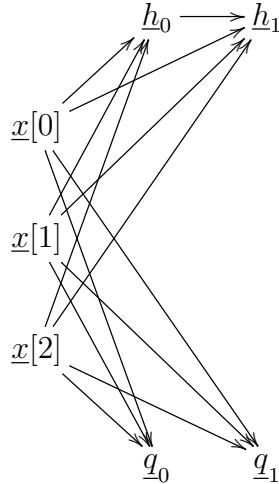


Figure 79.2: \underline{q} and \underline{h} have $nh = 2$ mirroring components and those of \underline{q} are independent at fixed \underline{x} .

Suppose $\underline{h} = (h_0, h_1, \dots, h_{nh-1})$ and $\underline{q} = (q_0, q_1, \dots, q_{nh-1})$ where $h_i \in S_{h_i}$ and $q_i \in S_{h_i}$ for all i . We say \underline{q} and \underline{h} have nh mirroring components and those of \underline{q} are independent at fixed \underline{x} if

$$P_{\underline{q}|\underline{x}}(\underline{h}|\vec{x}) = \prod_i P_{q_i|\underline{x}}(h_i|\vec{x}). \quad (79.2)$$

The bnet Fig.79.2 describes the scenario that we have in mind: The samples $x[\sigma]$ are i.i.d.. Each component h_i of \underline{h} has a mirroring component q_i in \underline{q} . The components of \underline{h} are correlated whereas those of \underline{q} are independent at fixed \underline{x} .

Claim 116 *If \underline{q} and \underline{h} have nh mirroring components and those of \underline{q} are independent at fixed \underline{x} and $\bar{D}_{KL}(P_{\underline{q}|\underline{x}} \| P_{\underline{h}|\underline{x}})$ is minimum over all $P_{\underline{q}|\underline{x}}$, then*

$$P_{q_i|\underline{x}}(q_i|\vec{x}) = \mathcal{N}(!q_i) e^{E_{(q_j)_{j \neq i}} [\ln P_{\underline{h}|\underline{x}}(\underline{h}=q|\vec{x})]} \quad (79.3)$$

$$= \mathcal{N}(!q_i) e^{E_{(q_j)_{j \neq i}} [\ln P_{\underline{h}, \underline{x}}(\underline{h}=q, \vec{x})]} \quad (79.4)$$

for all i .

proof:

Since all quantities in Eq.(79.3) are conditioned on \vec{x} , let us omit all mention of \vec{x} in this proof.

Let

$$\mathcal{L} = \mathcal{L}_0 + \mathcal{L}_1 \quad (79.5)$$

where

$$\mathcal{L}_0 = D_{KL}(P_{\underline{q}} \parallel P_{\underline{h}}) \quad (79.6)$$

$$= \sum_h P_{\underline{q}}(h) \ln \frac{P_{\underline{q}}(h)}{P_{\underline{h}}(h)} \quad (79.7)$$

$$= \sum_h P_{\underline{q}}(h) \ln P_{\underline{q}}(h) - \sum_h P_{\underline{q}}(h) \ln P_{\underline{h}}(h) \quad (79.8)$$

$$= \sum_i \sum_{h_i} P_{\underline{q}_i}(h_i) \ln P_{\underline{q}_i}(h_i) - \sum_h P_{\underline{q}}(h) \ln P_{\underline{h}}(h) \quad (79.9)$$

and

$$\mathcal{L}_1 = \sum_i \lambda_i \left[\sum_{h_i} P_{\underline{q}_i}(h_i) - 1 \right]. \quad (79.10)$$

Then

$$\delta \mathcal{L} = \sum_i \sum_{h_i} \delta P_{\underline{q}_i}(h_i) \left[\ln P_{\underline{q}_i}(h_i) + 1 + \lambda_i - \frac{1}{nh} \sum_{(h_j)_{j \neq i}} \prod_{(h_j)_{j \neq i}} \{P_{\underline{q}_j}(h_j)\} \ln P_{\underline{h}}(h) \right]. \quad (79.11)$$

Hence,

$$P_{\underline{q}_i}(h_i) = \mathcal{N}(!h_i) e^{\sum_{(h_j)_{j \neq i}} \left\{ \prod_{(h_j)_{j \neq i}} P_{\underline{q}_j}(h_j) \right\} \ln P_{\underline{h}}(h)}. \quad (79.12)$$

QED

Note that Eq.(79.3) yields a system of nh nonlinear equations in nh unknowns $(P_{\underline{q}_i|\vec{x}})_{i=0,1,\dots,nh-1}$. This system is usually solved recursively.

79.1 Free Energy $\mathcal{F}(\vec{x})$

To simplify the notation below, let us introduce the following abbreviations:

$$P(h|\vec{x}) = P_{\underline{h}|\vec{x}}(h|\vec{x}) \quad (79.13)$$

$$P(h, \vec{x}) = P_{\underline{h}, \vec{x}}(h, \vec{x}) \quad (79.14)$$

$$P(\vec{x}) = P_{\vec{x}}(\vec{x}) \quad (79.15)$$

Note that

$$D_{KL}(P_{\underline{q}|\vec{x}} \parallel P_{\underline{h}|\vec{x}}) = \sum_h P_{\underline{q}|\vec{x}}(h|\vec{x}) \ln \frac{P_{\underline{q}|\vec{x}}(h|\vec{x})}{P(h|\vec{x})} \quad (79.16)$$

$$= \sum_h P_{\underline{q}|\vec{x}}(h|\vec{x}) \ln \frac{P_{\underline{q}|\vec{x}}(h|\vec{x})}{P(h, \vec{x})} + \ln P(\vec{x}) \quad (79.17)$$

$$= \mathcal{F}(\vec{x}) + \ln P(\vec{x}) \quad (79.18)$$

Hence, the **Free energy** $\mathcal{F}(\vec{x})$ is defined as

$$\mathcal{F}(\vec{x}) = \sum_h P_{\underline{q}|\vec{x}}(h|\vec{x}) \ln \frac{P_{\underline{q}|\vec{x}}(h|\vec{x})}{P(h, \vec{x})} \quad (79.19)$$

$$= E_{\underline{q}|\vec{x}} \left[\ln \frac{P_{\underline{q}|\vec{x}}(q|\vec{x})}{P_{\underline{h},\vec{x}}(q, \vec{x})} \right]. \quad (79.20)$$

The name free energy is justified because

$$\mathcal{F}(\vec{x}) = \underbrace{- \sum_h P_{\underline{q}|\vec{x}}(h|\vec{x}) \ln P_{\underline{h},\vec{x}}(h, \vec{x})}_{U, \text{ Internal Energy}} + \underbrace{\sum_h P_{\underline{q}|\vec{x}}(h|\vec{x}) \ln P_{\underline{q}|\vec{x}}(h|\vec{x})}_{-S, \text{ minus Entropy}}. \quad (79.21)$$

It is also common to define a quantity called “ELBO” to be the negative of the free energy.

$$ELBO(\vec{x}) = -\mathcal{F}(\vec{x}) \quad (79.22)$$

ELBO stands for “Evidence Lower BOund”. That name is justified because

$$\underbrace{\ln P_{\vec{x}}(\vec{x})}_{\text{evidence} \leq 0} = \underbrace{D_{KL}(P_{\underline{q}|\vec{x}} \parallel P_{\underline{h}|\vec{x}})}_{\geq 0} - |ELBO(\vec{x})|. \quad (79.23)$$

Some properties of \mathcal{F} are:

- \mathcal{F} is non-negative.

$$\underbrace{D_{KL}(P_{\underline{q}|\vec{x}} \parallel P_{\underline{h}|\vec{x}})}_{\geq 0} + \underbrace{\ln \frac{1}{P_{\vec{x}}(\vec{x})}}_{\geq 0} = \mathcal{F}(\vec{x}) \quad (79.24)$$

- KL divergence is min iff \mathcal{F} is min at fixed $P(\vec{x})$.

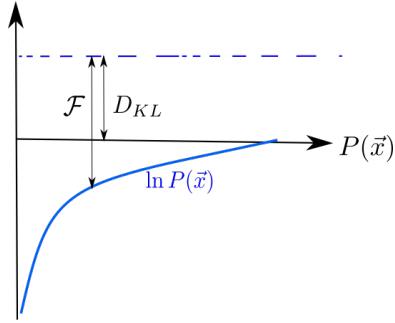


Figure 79.3: $D_{KL} + \ln \frac{1}{P(\vec{x})} = \mathcal{F}$.

During a variation δ that holds $P(\vec{x})$ fixed, the KL divergence and \mathcal{F} change by the same amount:

$$\delta D_{KL}(P_{g|\vec{x}} \parallel P_{h|\vec{x}}) = \delta \mathcal{F}(\vec{x}) \quad (79.25)$$

Chapter 80

XGBoost

This paper is based on the original XGBoost paper Ref.[4] and the excellent StatQuest videos (highly recommended) [52].

Extreme Gradient Boosting (XGBoost) (Chen, Guestrin, 2016) improves gradient boosting (Friedman, 1999) in a number of ways, such as by using a quadratic rather than linear approximation for the variational function.

In XGBoost, one calculates a sequence of functions where each function tries to correct the errors of the previous function. Then a series (i.e., linear combination) of those functions yields an estimate \hat{y}^σ of the target attribute y^σ . As will be shown in this chapter, XGBoost can be used in two cases: Regression (continuous target attribute) and Binary Classification (binary target attribute)¹. An implementation of the XGBoost algorithm is available as open source. It's written in C++, with Python and R interfaces.

Boosting (see this chapter on XGBoost and Chapter 1 on AdaBoost) and bagging (see Chapter 58 on Random Forest) are two methods of building a classifier function from an ensemble of classifier functions. These two methods are most commonly applied to dtrees: Boosting for an ensemble of small dtrees, and Bagging for a random forest (which is an ensemble of dtrees that are usually much more complicated than small dtrees).

80.1 Divergences

To set up a cost function for XGBoost, we begin by defining 2 types of “divergences” and calculating the first and second derivatives of those divergences:

- **Divergence for regression** (i.e., continuous classification, continuous target attribute). For $x, y \in \mathbb{R}$

$$D_{reg}(x, y) = \frac{1}{2}(x - y)^2 \quad (80.1)$$

¹XGBoost can also be used for classification into more than two classes, as long as a suitable Divergence function can be defined.

- **Divergence for binary classification** (i.e., binary target attribute). For $p, q \in [0, 1]$.

$$D_{bc}(p, q) = -\{p \ln q + (1-p) \ln(1-q)\} \quad (80.2)$$

$$= CE(\{p, 1-p\} \| \{q, 1-q\}) \quad (80.3)$$

The cross-entropy $CE()$ is defined in Chapter Notational Conventions and Preliminaries.

Claim 117

$$\partial_{\hat{y}} D_{reg}(y, \hat{y}) = \hat{y} - y \quad (80.4)$$

$$\partial_{\hat{y}}^2 D_{reg}(y, \hat{y}) = 1 \quad (80.5)$$

$$D(y, \hat{y} + h) \approx D(y, \hat{y}) + (\hat{y} - y)h + \frac{1}{2}h^2 \quad (80.6)$$

proof: Obvious.

QED

Recall from section 0.21 in Chapter Notational Conventions and Preliminaries that

$$\hat{y} = \text{smoid}(\text{lodds}(\hat{y})) \quad (80.7)$$

Let us abbreviate $s() = \text{smoid}()$ and $l() = \text{lodds}()$ so

$$\hat{y} = s(l(\hat{y})). \quad (80.8)$$

Claim 118 ²

$$\partial_l D(y, \hat{y}(l)) = \hat{y} - y \quad (80.9)$$

$$\partial_l^2 D(y, \hat{y}(l)) = \hat{y}(1 - \hat{y}) \quad (80.10)$$

$$D(y, \hat{y}(l + \Delta l)) \approx D(y, \hat{y}(l)) + (\hat{y} - y)\Delta l + \frac{1}{2}\hat{y}(1 - \hat{y})(\Delta l)^2 \quad (80.11)$$

²Note that $D_{bc}(y, \hat{y}) \neq D_{bc}(\hat{y}, y)$; i.e., D_{bc} is not symmetric in its two arguments. Normally $0 < \hat{y} < 1$ and $y \in \{0, 1\}$. Since the log is applied only to the second argument, to avoid logs of zero, it is better to have \hat{y} as the second argument.

proof:

$$\frac{\partial D(y, s)}{\partial s} = -\partial_s \{y \ln s + (1 - y) \ln(1 - s)\} \quad (80.12)$$

$$= -\frac{y}{s} + \frac{1 - y}{1 - s} \quad (80.13)$$

$$= \frac{-y(1 - s) + (1 - y)s}{s(1 - s)} \quad (80.14)$$

$$= \frac{-y + s}{s(1 - s)} \quad (80.15)$$

Recall that $\text{smoid}'(l) = \text{smoid}(l)[1 - \text{smoid}(l)]$ so

$$\frac{\partial s}{\partial l} = s(1 - s). \quad (80.16)$$

$$\frac{\partial D(y, s)}{\partial l} = \frac{\partial s}{\partial l} \frac{\partial D(y, s)}{\partial s} = -y + s = -y + \hat{y} \quad (80.17)$$

$$\frac{\partial^2 D(y, s)}{\partial l^2} = \frac{\partial s}{\partial l} = s(1 - s) \quad (80.18)$$

QED

80.2 Minimizing Cost function for single tree

Let

$\sigma \in \Sigma$ be an individual in a population Σ

$x^\sigma \in S_{\underline{x}}$ be a feature vector $x^\sigma = (x_i^\sigma)_{i=0,1,\dots,n_f-1}$

$t \in \{0, 1, \dots, nt - 1\}$ be the tree index

\mathcal{L}_t be the set of leafs of tree t

$\ell \in \mathcal{L}_t$ be a leaf in tree t

$w_t^\ell \in \mathbb{R}$

$f_t : S_{\underline{x}} \rightarrow \mathbb{R}$

$\ell_t : \Sigma \rightarrow \mathcal{L}_t$, $\ell_t(\Sigma) = \mathcal{L}_t$.

$\ell_t(\sigma)$ be the leaf of individual σ in tree t

$\Sigma_t^\ell = \{\sigma \in \Sigma : \ell_t(\sigma) = \ell\}$

Define the function f_t by

$$f_t(x^\sigma) = \sum_{\ell \in \mathcal{L}_t} w_t^\ell \mathbb{1}(\sigma \in \Sigma_t^\ell) = w_t^{\ell_t(\sigma)}. \quad (80.19)$$

$f_t(x^\sigma)$ gives the output value for tree t and feature vector x^σ .

Define the **cost function** for tree t as follows

$$\mathcal{C}_t = \sum_{\sigma} D(\hat{y}_t^{\sigma}, y^{\sigma}) + \underbrace{\sum_{\ell \in \mathcal{L}_t} \left[\gamma + \frac{\lambda}{2} (w_t^{\ell})^2 \right]}_{\text{regulator}} , \quad (80.20)$$

where $\gamma > 0, \lambda > 0$ are regulator parameters.

The estimate \hat{y}_t^{σ} , using trees from 0 to t , of the target attribute y^{σ} , is defined by

$$\hat{y}_0^{\sigma} = f_0(x^{\sigma}) = \text{arbitrary constant, XGBoost default for this is 0.5} \quad (80.21)$$

$$\hat{y}_1^{\sigma} = f_0(x^{\sigma}) + f_1(x^{\sigma}) \quad (80.22)$$

$$\hat{y}_2^{\sigma} = f_0(x^{\sigma}) + f_1(x^{\sigma}) + f_2(x^{\sigma}) \quad (80.23)$$

$$\vdots \quad (80.24)$$

$$\hat{y}_t^{\sigma} = \sum_{t' \leq t} f_{t'}(x^{\sigma}) = \hat{y}_{t-1}^{\sigma} + f_t(x^{\sigma}) \quad (80.25)$$

In the cost function \mathcal{C}_t , we approximate the divergence $D()$ by a second order Taylor approximation:

$$D(y^{\sigma}, \hat{y}_t^{\sigma}) = D(y^{\sigma}, \hat{y}_{t-1}^{\sigma} + \underbrace{f_t(x^{\sigma})}_{\delta}) \quad (80.26)$$

$$\approx D(y^{\sigma}, \hat{y}_{t-1}^{\sigma}) + a_t^{\sigma} \delta + \frac{1}{2} b_t^{\sigma} \delta^2 \quad (80.27)$$

$$= D(y^{\sigma}, \hat{y}_{t-1}^{\sigma}) + a_t^{\sigma} w_t^{\ell_t(\sigma)} + \frac{1}{2} b_t^{\sigma} (w_t^{\ell_t(\sigma)})^2 , \quad (80.28)$$

where

$$a_t^{\sigma} = [\partial_{\hat{y}} D(y^{\sigma}, \hat{y})]_{\hat{y}=\hat{y}_{t-1}^{\sigma}} , \quad (80.29)$$

$$b_t^{\sigma} = [\partial_{\hat{y}}^2 D(y^{\sigma}, \hat{y})]_{\hat{y}=\hat{y}_{t-1}^{\sigma}} . \quad (80.30)$$

a_t^{σ} is called g for gradient and b_t^{σ} is called h for Hessian in Ref.[4]. Table 80.1 gives the values for a_t^{σ} and b_t^{σ} for Regression (reg) and Binary classification (bc).

Define the **residual** for tree t and individual σ by

$$r_t^{\sigma} = y^{\sigma} - \hat{y}_{t-1}^{\sigma} . \quad (80.31)$$

Note that

$$\sum_{\sigma} = \sum_{\ell \in \mathcal{L}_t} \sum_{\sigma \in \Sigma_t^{\ell}} . \quad (80.32)$$

	Regression ($y^\sigma \in \mathbb{R}$, $\hat{y}_t^\sigma \in \mathbb{R}$, $D = D_{reg}$)	Binary Classification ($y^\sigma \in \{0, 1\}$, $\hat{y}_t^\sigma \in [0, 1]$, $D = D_{bc}$)
a_t^σ	$\hat{y}_{t-1}^\sigma - y^\sigma = \text{neg.residual}$	$\hat{y}_{t-1}^\sigma - y^\sigma = \text{neg.residual}$
b_t^σ	1	$\hat{y}_{t-1}^\sigma(1 - \hat{y}_{t-1}^\sigma)$

Table 80.1: The first (a_t^σ) and second (b_t^σ) derivatives for Regression (reg) and Binary classification (bc).

Define

$$A_t^\ell = \sum_{\sigma \in \Sigma_t^\ell} a_t^\sigma \quad (80.33)$$

and

$$B_t^\ell = \sum_{\sigma \in \Sigma_t^\ell} b_t^\sigma. \quad (80.34)$$

Now we can rewrite the cost function as

$$\mathcal{C}_t = \underbrace{\sum_{\sigma} D(\hat{y}_{t-1}^\sigma, y^\sigma)}_{\mathcal{K}} + \sum_{\ell \in \mathcal{L}_t} \left[A_t^\ell w_t^\ell + \frac{1}{2} B_t^\ell (w_t^\ell)^2 \right] + \sum_{\ell \in \mathcal{L}_t} \left[\gamma + \frac{\lambda}{2} (w_t^\ell)^2 \right] \quad (80.35)$$

$$= \sum_{\ell \in \mathcal{L}_t} \left[\gamma + A_t^\ell w_t^\ell + \frac{1}{2} (B_t^\ell + \lambda) (w_t^\ell)^2 \right] \quad (\text{absorbed } \mathcal{K} \text{ into } \gamma) \quad (80.36)$$

The cost function \mathcal{C}_t can be minimized over w_t^ℓ :

$$0 = \delta \mathcal{C}_t = \sum_{\ell \in \mathcal{L}_t} \delta w_t^\ell [A_t^\ell + (B_t^\ell + \lambda) w_t^\ell] \quad (80.37)$$

Therefore, the cost is minimized when

$$w_t^\ell = \frac{-A_t^\ell}{B_t^\ell + \lambda}. \quad (80.38)$$

The optimum cost is

$$\mathcal{C}_t = \sum_{\ell \in \mathcal{L}_t} \underbrace{\left[\gamma - \frac{(A_t^\ell)^2}{2(B_t^\ell + \lambda)} \right]}_{C_t^\ell} \quad (80.39)$$

SS_t^ℓ is called the **similarity score** for leaf ℓ and tree t . Note that an increase in similarity decreases the cost.

80.3 Leaf Splitting

Suppose leaf ℓ_0 splits into leafs ℓ_L and ℓ_R . Then

$$\Sigma_t^{\ell_0} = \Sigma_t^{\ell_L} \cup \Sigma_t^{\ell_R}, \quad \Sigma_t^{\ell_L} \cap \Sigma_t^{\ell_R} = \emptyset \quad (80.40)$$

so

$$A_t^{\ell_0} = A_t^{\ell_L} + A_t^{\ell_R} \quad (80.41)$$

and

$$B_t^{\ell_0} = B_t^{\ell_L} + B_t^{\ell_R}. \quad (80.42)$$

Hence,

$$\mathcal{C}_t^{\ell_j} = \gamma - \frac{(A_t^{\ell_j})^2}{2(B_t^{\ell_j} + \lambda)} \quad \text{for } j = L, R \quad (80.43)$$

$$\mathcal{C}_t^{\ell_0} = \gamma - \frac{(A_t^{\ell_L} + A_t^{\ell_R})^2}{2(B_t^{\ell_L} + B_t^{\ell_R} + \lambda)} \quad (80.44)$$

Define the **XGBoost branch Gain** for a binary tree branch by

$$Gain_{XGB} = \mathcal{C}_t^{\ell_0} - \mathcal{C}_t^{\ell_L} - \mathcal{C}_t^{\ell_R} \quad (80.45)$$

$$= \underbrace{\left\{ SS_t^{\ell_L} + SS_t^{\ell_R} - SS_t^{\ell_0} \right\}}_{\Delta SS_t} - \gamma \quad (80.46)$$

By splitting a leaf, we create a new binary branch with 2 new leafs. We successively add new branches to the tree until some stopping criterion is satisfied. We continue adding branches to the tree as long as they have a positive gain, and as long as the number of levels is smaller than an upper bound input parameter (XGBoost's default maximum number of tree levels is six). XGBoost also rules out leafs that have a denominator quantity B_t^ℓ (called the **cover**) smaller than a lower bound input parameter. This branching process is illustrated in Figs.80.1 and 80.2.

80.4 Pruning

If we raise γ or raise λ , branches that previously had positive gain may acquire a negative gain. Get rid of branches from highest level that now have negative gain. This will generate new branches. Get rid of new branches from the highest level that have negative gain. Continue this process until all highest level branches have positive gain. This may reduce a tree to a single node or even rule out the entire tree.

λ measures **level of insensitivity** to observations, and γ measures **level of tree simplicity**.

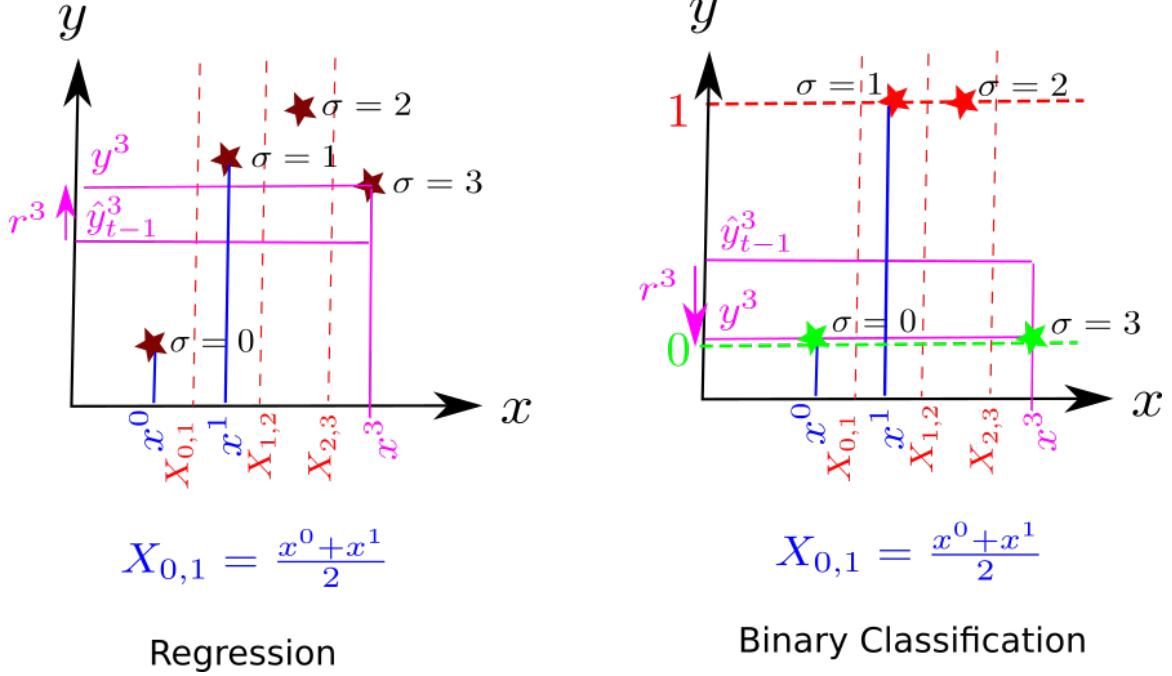
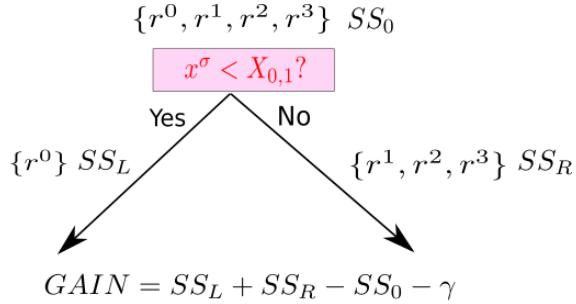


Figure 80.1: Plot of target attribute y^σ versus a feature vector x^σ with a single component. More generally, the feature vector $x^\sigma = (x_j^\sigma)_{j=0,1,\dots,nf-1}$ can have multiple components called features or attributes. The left plot refers to a population of 4 individuals and regression so $y^\sigma, \hat{y}^\sigma \in \mathbb{R}$. The right plot refers to a population of 4 individuals and binary classification so $y^\sigma \in \{0, 1\}, \hat{y}^\sigma \in [0, 1]$. $X_{j,j+1}$ for $j = 0, 1, 2$ is the average between 2 adjacent values of x^σ .



Repeat for $x^\sigma < X_{1,2}$? and $x^\sigma < X_{2,3}$?
Choose inequality with highest GAIN

Figure 80.2: This figure refers to the situation of Fig.80.1. Out of all allowed tree branches, we choose the one with the highest gain.

80.5 Feature Binning

For really large population sizes $|\Sigma|$, it is convenient to bin the set $\{x_i^\sigma : \sigma \in \Sigma\}$ for each i .

A common bin type is quantiles. Quantiles are bins $[X_{j-1}, X_j]$ for $j = 0, 1, \dots, nbins - 1$ that all contain approximately the same number of points. For example, in Fig.80.2, with 1 point quantile bins, use $[X_{j-1}, X_j]$ for $j = 0, 1, 2, 3$. With 2 point quantile bins, use $[X_{j-1}, X_j]$ for $j = 0, 1$.

Use the right edge of each bin as the X_j in the question $x^\sigma < X_j?$, and choose the question which yields the highest gain.

80.6 Final estimate of target attribute

In this section, we will give a formula for the final estimate of the target attribute. Previously, we set the learning rate η to one. Here we restore η to an arbitrary value between 0 and 1.

Instead of using

$$f(x^\sigma) = \sum_{t=0}^{nt-1} f_t(x^\sigma), \quad (80.47)$$

we will use

$$f(x^\sigma) = \sum_{t=0}^{nt-1} (\eta)^t f_t(x^\sigma), \quad (80.48)$$

where $\eta \in [0, 1]$ is called the **learning rate**. This has the effect of compensating for the f_t 's with $t > nt - 1$ that would have been included had we used an infinite series. Also, think of Eq.(80.47) as an approximation (a truncated Taylor expansion in powers of Δx) of a function $f(x + \Delta x)$, and think of Eq.(80.48) as an analogous approximation of the function $f(x + \eta \Delta x)$.

For the case of Regression, we get:

$$f(x^\sigma) = \sum_t (\eta)^t \left(\frac{-A_t^{\ell_t(\sigma)}}{B_t^{\ell_t(\sigma)} + \lambda} \right). \quad (80.49)$$

For the case of Binary Classification, we get

$$f(x^\sigma) = \underbrace{\sum_{t=0}^{nt-1} (\eta)^t \text{smoid} \left(\frac{-A_t^{\ell_t(\sigma)}}{B_t^{\ell_t(\sigma)} + \lambda} \right)}_{\substack{\text{lodds(probability)} \\ \text{probability}}}. \quad (80.50)$$

80.7 Bnet for XGBoost

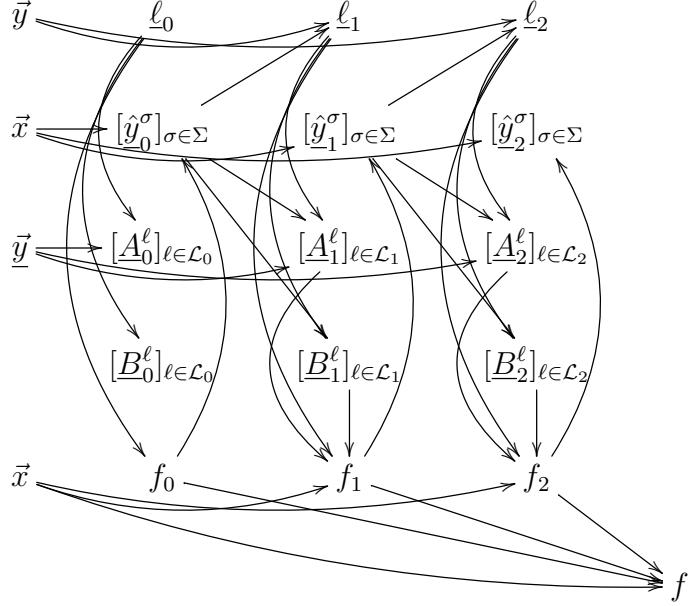


Figure 80.3: Bnet for XGBoost assuming $nt = 3$. Nodes that appear multiple times (namely \vec{x} and \vec{y}) should be considered the same node, drawn multiple times for clarity. The parameters λ, γ, η are taken to be global. Recall that $\ell_t : \Sigma \rightarrow \mathcal{L}_t$. From the function $\ell_t()$, we can derive its range $\mathcal{L}_t = \{\ell_t(\sigma) : \sigma \in \Sigma\}$, and $\Sigma_t^\ell = \{\sigma \in \Sigma : \ell_t(\sigma) = \ell\}$ for all $\ell \in \mathcal{L}_t$.

Fig.80.3 gives our bnet for the XGBoost algo assuming $nt = 3$. The TPMs, printed in blue, for this bnet, are as follows:

For $t = 0$, ℓ_0 describes a single node “tree” such that $\ell_0(x^\sigma) = f_0(x^\sigma) = 0.5$ for all $\sigma \in \Sigma$. For $t \geq 1$,

$$P(\ell_t | \vec{y}, [\hat{y}_{t-1}^\sigma]_\sigma) = \begin{array}{l} \text{build tree } \ell_t \text{ using } y^\sigma \text{ and } \hat{y}_{t-1}^\sigma \text{ for all } \sigma \in \Sigma. \\ \lambda \text{ and } \gamma \text{ are used here.} \end{array} \quad (80.51)$$

$$P([\hat{y}_t^\sigma]_\sigma | f_t, \vec{x}) = \prod_{\sigma \in \Sigma} \mathbb{1}(\hat{y}_t^\sigma = f_t(x^\sigma)) \quad (80.52)$$

$$P([A_t^\ell]_{\ell \in \mathcal{L}_t} | \vec{y}, [\hat{y}_{t-1}^\sigma]_\sigma, \ell_t) = \prod_{\ell \in \mathcal{L}_t} \mathbb{1}(A_t^\ell = - \sum_{\sigma \in \Sigma_t^\ell} \underbrace{(y^\sigma - \hat{y}_{t-1}^\sigma)}_{r_t^\sigma}) \quad (80.53)$$

$$P(B_t^\ell | \ell \in \mathcal{L}_t | [\hat{y}_{t-1}^\sigma]_\sigma, \ell_t) = \prod_{\ell \in \mathcal{L}_t} \mathbb{1}(B_t^\ell = \sum_{\sigma \in \Sigma_t^\ell} \left\{ \begin{array}{ll} 1 & \text{if reg.} \\ \hat{y}_{t-1}^\sigma (1 - \hat{y}_{t-1}^\sigma) & \text{if b.c.} \end{array} \right\}) \quad (80.54)$$

$$P(f_t | [A_t^\ell]_{\ell \in \mathcal{L}_t}, [B_t^\ell]_{\ell \in \mathcal{L}_t}, \ell_t, \vec{x}) = \prod_{\sigma \in \Sigma} \mathbb{1} \left(f_t(x^\sigma) = \left\{ \begin{array}{ll} 1 & \text{if reg.} \\ \text{smoid} & \text{if b.c.} \end{array} \right\} \frac{-A^{\ell_t(\sigma)}}{2(B^{\ell_t(\sigma)} + \lambda)} \right) \quad (80.55)$$

$$P(f | [f_t]_t, \vec{x}) = \prod_{\sigma \in \Sigma} \mathbb{1}(f(x^\sigma) = \sum_t (\eta)^t f_t(x^\sigma)) \quad (80.56)$$

Chapter 81

Zero Information Transmission (Graphoid Axioms)

This chapter assumes that you have read Chapter 19 on d-separation.

The following quantities play a very prominent role in the d-separation Theorem that we enunciated in Chapter 19.

- the mutual information (MI)
(aka information transmission) $H(\underline{a} : \underline{b})$
- the conditional mutual information (CMI)
(aka conditional information transmission) $H(\underline{a} : \underline{b} | \underline{c})$

MI can be viewed as the special case of CMI, when the set of variables being conditioned on is empty. Particularly prominent in d-separation discussions are probability distributions for which CMI vanishes. The goal of this chapter is to study such probability distributions.

Recall that CMI is non-negative and symmetric in its first two variables (i.e., $H(\underline{a} : \underline{b} | \underline{c}) = H(\underline{b} : \underline{a} | \underline{c})$). Another very useful property of CMI is its chain rule

Claim 119 (*Chain Rule for CMI*)

$$H(\underline{y} : \underline{x}^n) = \sum_i H(\underline{y} : \underline{x}_i | \underline{x}_{<i}), \quad (81.1)$$

where $\underline{x}^n = (\underline{x}_0, \underline{x}_1, \dots, \underline{x}_{n-1})$ and $\underline{x}_{<i} = (\underline{x}_0, \underline{x}_1, \dots, \underline{x}_{i-1})$.

proof:

$$\frac{P(y|x_{<i+1})}{P(y)} = \frac{P(y|x_i, x_{<i})}{P(y|x_{<i})} \cdot \frac{P(y|x_{<i})}{P(y)} \quad (81.2)$$

Therefore,

$$\ln \frac{P(y|x_{<i+1})}{P(y)} = \ln \frac{P(y|x_i, x_{<i})}{P(y|x_{<i})} + \ln \frac{P(y|x_{<i})}{P(y)} \quad (81.3)$$

If we now apply $\sum_{y,x^n} P(y, x^n)$ to both sides, we get

$$H(\underline{y} : \underline{x}_{<i+1}) = H(\underline{y} : \underline{x}_i | \underline{x}_{<i}) + H(\underline{y} : \underline{x}_{<i}) \quad (81.4)$$

QED

A trivial but very useful consequence of the chain rule for CMI is:

$$H(\underline{y} : \underline{x}^n) = 0 \iff H(\underline{y} : \underline{x}_i | \underline{x}_{<i}) = 0 \text{ for all } i. \quad (81.5)$$

81.1 Consequences of Eq.(81.5)

Table 81.1 gives a set of statements about CMI referred to as the Graphoid Axioms in chapter 1 of Ref.[39]. See Ref.[39] to learn the history of these axioms. The purpose of this section is to prove that the graphoid axioms are all a simple consequence of Eq.(81.5).

Symmetry	$\underline{a} \perp_P \underline{b} \implies \underline{b} \perp_P \underline{a}$ $H(\underline{a} : \underline{b}) = 0 \implies H(\underline{b} : \underline{a}) = 0$
Decomposition	$\underline{a} \perp_P \underline{b}, \underline{c} \implies \underline{a} \perp_P \underline{b} \text{ and } \underline{a} \perp_P \underline{c}$ $H(\underline{a} : \underline{b}, \underline{c}) = 0 \implies H(\underline{a} : \underline{b}) = 0 \text{ and } H(\underline{a} : \underline{c}) = 0$
Weak Union	$\underline{a} \perp_P \underline{b}, \underline{c} \implies \underline{a} \perp_P \underline{b} \underline{c} \text{ and } \underline{a} \perp_P \underline{c} \underline{b}$ $H(\underline{a} : \underline{b}, \underline{c}) = 0 \implies H(\underline{a} : \underline{b} \underline{c}) = 0 \text{ and } H(\underline{a} : \underline{c} \underline{b}) = 0$
Contraction	$\underline{a} \perp_P \underline{b} \underline{c} \text{ and } \underline{a} \perp_P \underline{c} \implies \underline{a} \perp_P \underline{b}, \underline{c}$ $H(\underline{a} : \underline{b} \underline{c}) = 0 \text{ and } H(\underline{a} : \underline{c}) = 0 \implies H(\underline{a} : \underline{b}, \underline{c}) = 0$
Intersection	$\underline{a} \perp_P \underline{b} \underline{c}, \underline{d} \text{ and } \underline{a} \perp_P \underline{d} \underline{c}, \underline{b} \implies \underline{a} \perp_P \underline{b}, \underline{d} \underline{c}$ $H(\underline{a} : \underline{b} \underline{c}, \underline{d}) = 0 \text{ and } H(\underline{a} : \underline{d} \underline{c}, \underline{b}) = 0 \implies H(\underline{a} : \underline{b}, \underline{d} \underline{c}) = 0$

Table 81.1: Graphoid Axioms

Claim 120 *Table 81.1 is true.*

proof:

- **Symmetry**

Follows trivially from $H(\underline{a} : \underline{b}) = H(\underline{b} : \underline{a})$.

- **Decomposition**

From the chain rule for CMI, we have

$$H(\underline{a} : \underline{b}, \underline{c}) = H(\underline{a} : \underline{b}|\underline{c}) + H(\underline{a} : \underline{c}), \quad (81.6)$$

and

$$H(\underline{a} : \underline{b}, \underline{c}) = H(\underline{a} : \underline{c} | \underline{b}) + H(\underline{a} : \underline{b}) . \quad (81.7)$$

Hence,

$$H(\underline{a} : \underline{b}, \underline{c}) = 0 \quad (81.8)$$

implies

$$H(\underline{a} : \underline{b} | \underline{c}) = H(\underline{a} : \underline{c}) = 0 , \quad (81.9)$$

and

$$H(\underline{a} : \underline{c} | \underline{b}) = H(\underline{a} : \underline{b}) = 0 . \quad (81.10)$$

- **Weak Union**

Already proven in proof of Decomposition.

- **Contraction**

From chain rule for CMI, we have

$$H(\underline{a} : \underline{b}, \underline{c}) = H(\underline{a} : \underline{b} | \underline{c}) + H(\underline{a} : \underline{c}) . \quad (81.11)$$

- **Intersection**

From the chain rule for CMI, we have

$$H(\underline{a} : \underline{b}, \underline{d} | \underline{c}) = H(\underline{a} : \underline{b} | \underline{d}, \underline{c}) + H(\underline{a} : \underline{d} | \underline{c}) , \quad (81.12)$$

and

$$H(\underline{a} : \underline{b}, \underline{d} | \underline{c}) = H(\underline{a} : \underline{d} | \underline{b}, \underline{c}) + H(\underline{a} : \underline{b} | \underline{c}) . \quad (81.13)$$

Thus,

$$H(\underline{a} : \underline{b}, \underline{d} | \underline{c}) = 0 \quad (81.14)$$

implies

$$H(\underline{a} : \underline{b} | \underline{d}, \underline{c}) = H(\underline{a} : \underline{d} | \underline{c}) = 0 , \quad (81.15)$$

and

$$H(\underline{a} : \underline{d} | \underline{b}, \underline{c}) = H(\underline{a} : \underline{b} | \underline{c}) = 0 . \quad (81.16)$$

QED

Bibliography

- [1] Elias Bareinboim, Jin Tian, and Judea Pearl. Recovering from selection bias in causal and statistical inference. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014. https://ftp.cs.ucla.edu/pub/stat_ser/r425.pdf.
- [2] Dan Bendel. Metropolis-Hastings: A comprehensive overview and proof. <https://similarweb.engineering/mcmc/>.
- [3] Alexandra M Carvalho. Scoring functions for learning Bayesian networks. http://www.lx.it.pt/~asmc/pub/talks/09-TA/ta_pres.pdf.
- [4] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. <https://arxiv.org/abs/1603.02754>.
- [5] Carlos Cinelli, Andrew Forney, and Judea Pearl. A crash course in good and bad controls. https://ftp.cs.ucla.edu/pub/stat_ser/r493.pdf.
- [6] Scott Cunningham. *Causal inference: The mixtape*. Yale University Press, 2021. <https://mixtape.scunning.com/index.html>.
- [7] Robin J. Evans. Graphical methods for inequality constraints in marginalized DAGs. <https://arxiv.org/abs/1209.2978>.
- [8] Matheus Facure Alves. *Causal Inference for The Brave and True*. 2021. <https://matheusfacure.github.io/python-causality-handbook/landing-page.html>.
- [9] George Fei. Modeling uplift directly: Uplift decision tree with kl divergence and euclidean distance as splitting criteria. <https://www.aboutwayfair.com/tech-innovation/modeling-uplift-directly-uplift-decision-tree-with-kl-divergence-and-euclidean-distance-as-splitting-criteria>.
- [10] Charles Fox, Neil Girdhar, and Kevin Gurney. A causal Bayesian network view of reinforcement learning. <https://www.aaai.org/Papers/FLAIRS/2008/FLAIRS08-030.pdf>.
- [11] Bruno Gonçalves. Model testing and causal search. blog post <https://medium.com/data-for-science/causal-inference-part-vii-model-testing-and-causal-search-536b796f0384>.

- [12] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, David Warde-Farley, Bing Xu, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. <https://arxiv.org/abs/1406.2661>.
- [13] Pierre Gutierrez and Jean-Yves Gérardy. Causal inference and uplift modelling: A review of the literature. In *Proceedings of The 3rd International Conference on Predictive Applications and APIs*, pages 1–13, 2017. <http://proceedings.mlr.press/v67/gutierrez17a.html>.
- [14] James Douglas Hamilton. *Time series analysis*. Princeton university press, 2020.
- [15] Christina Heinze-Deml. Causality, spring semester 2019 at ETH Zurich. https://stat.ethz.ch/lectures/ss19/causality.php#course_materials.
- [16] Cecil Huang and Adnan Darwiche. Inference in belief networks: A procedural guide. *International journal of approximate reasoning*, 15(3):225–263, 1996. <http://www.artiste.com/Huang-Darwiche1996.pdf>.
- [17] Michael I. Jordan. course: Stat260-Bayesian modeling and inference, lecture date: February 8-2010, title: The conjugate prior for the Normal distribution. <https://people.eecs.berkeley.edu/~jordan/courses/260-spring10/lectures/lecture5.pdf>.
- [18] Chaitanya K. Joshi. Transformer (machine learning model). <https://graphdeeplearning.github.io/post/transformers-are-gnns/>.
- [19] Chung-Ming Kuan. Introduction to time series analysis, Fall 2014 lectures given at the Department of Finance, National Taiwan University. https://homepage.ntu.edu.tw/~ckuan/pdf/2014fall/Lec-TimeSeries_slide-Fall2014.pdf.
- [20] Steffen L Lauritzen and David J Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 50(2):157–194, 1988. <http://www.eecis.udel.edu/~shatkay/Course/papers/Lauritzen1988.pdf>.
- [21] Sergey Levine. Course CS 285 at UC Berkeley, Deep reinforcement learning. <http://rail.eecs.berkeley.edu/deeprlcourse/>.
- [22] Ang Li. Ph.D. Thesis, UCLA 2021. https://ftp.cs.ucla.edu/pub/stat_ser/r507.pdf.
- [23] Dimitris Margaritis. Learning Bayesian network model structure from data (thesis, 2003, Carnegie Mellon Univ). <https://apps.dtic.mil/sti/citations/ADA461103>.

- [24] Adam A Margolin, Ilya Nemenman, Katia Basso, Chris Wiggins, Gustavo Stolovitzky, Riccardo Dalla Favera, and Andrea Califano. Aracne: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. In *BMC bioinformatics*, volume 7, page S7. Springer, 2006. <https://link.springer.com/article/10.1186/1471-2105-7-S1-S7>.
- [25] Samuele Mazzanti. Black-box models are actually more explainable than a logistic regression. <https://towardsdatascience.com/black-box-models-are-actually-more-explainable-than-a-logistic-regression-f263c22795d>.
- [26] Samuele Mazzanti. SHAP values explained exactly how you wished someone explained to you. <https://towardsdatascience.com/shap-explained-the-way-i-wish-someone-explained-it-to-me-ab81cc69ef30>.
- [27] Robert J. McEliece, David J. C. MacKay, and Jung-Fu Cheng. Turbo decoding as an instance of Pearl’s belief propagation algorithm. <http://authors.library.caltech.edu/6938/1/MCEieeejstc98.pdf>.
- [28] Scott Mueller, Ang Li, and Judea Pearl. Causes of effects: Learning individual responses from population data. *arXiv preprint arXiv:2104.13730*, 2021. <https://arxiv.org/abs/2104.13730>.
- [29] Brady Neal. Introduction to causal inference, Fall 2020, lectures and book. <https://www.bradyneal.com/causal-inference-course>.
- [30] Richard E Neapolitan. *Learning Bayesian networks*. Pearson Prentice Hall, 2004.
- [31] Andrew Ng. Lecture at deeplearning.ai on recurrent neural networks. <http://www.ar-tiste.com/ng-lec-rnn.pdf>.
- [32] Gregory Nuel. Tutorial on exact belief propagation in Bayesian networks: from messages to algorithms. <https://arxiv.org/abs/1201.4724>.
- [33] Judea Pearl. Linear models: A useful microscope for causal analysis. https://ftp.cs.ucla.edu/pub/stat_ser/r409-corrected-reprint.pdf.
- [34] Judea Pearl. Mediating instrumental variables. https://ftp.cs.ucla.edu/pub/stat_ser/r210.pdf.
- [35] Judea Pearl. On the testability of causal models with latent and instrumental variables. <https://arxiv.org/abs/1302.4976>.
- [36] Judea Pearl. Reverend Bayes on inference engines: A distributed hierarchical approach. <https://www.aaai.org/Papers/AAAI/1982/AAAI82-032.pdf>, 1982.

- [37] Judea Pearl. *Probabilistic Inference in Intelligent Systems*. Morgan Kaufmann, 1988.
- [38] Judea Pearl. The causal mediation formula—a guide to the assessment of pathways and mechanisms. *Prevention science*, 13(4):426–436, 2012. <https://apps.dtic.mil/sti/pdfs/ADA557663.pdf>.
- [39] Judea Pearl. *Causality: Models, Reasoning, and Inference, Second Edition*. Cambridge University Press, 2013.
- [40] Judea Pearl. Causal and counterfactual inference. *The Handbook of Rationality*, pages 1–41, 2019. https://ftp.cs.ucla.edu/pub/stat_ser/r485.pdf.
- [41] Judea Pearl and Elias Bareinboim. Transportability of causal and statistical relations: A formal approach. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 25, 2011. <https://ojs.aaai.org/index.php/AAI/article/view/7861>.
- [42] Judea Pearl, Madelyn Glymour, and Nicholas P Jewell. *Causal inference in statistics: A primer*. John Wiley & Sons, 2016.
- [43] Judea Pearl and Dana Mackenzie. *The book of why: the new science of cause and effect*. Basic Books, 2018.
- [44] Judea Pearl and James M Robins. Probabilistic evaluation of sequential plans from causal models with hidden variables. *arXiv preprint arXiv:1302.4977*, 2013. <https://arxiv.org/abs/1302.4977>.
- [45] Ashwin Rao and Tikhon Jelvis. *Foundations of Reinforcement Learning with Applications in Finance*. <https://stanford.edu/~ashlearn/RLForFinanceBook/book.pdf>.
- [46] ReliaSoft. System analysis reference. http://reliawiki.org/index.php/System_Analysis_Reference.
- [47] Piotr Rzepakowski and Szymon Jaroszewicz. Decision trees for uplift modeling with single and multiple treatments. *Knowledge and Information Systems*, 32(2):303–327, 2012. <https://link.springer.com/content/pdf/10.1007/s10115-011-0434-0.pdf>.
- [48] Scholarpedia. Granger causality. http://www.scholarpedia.org/article/Granger_causality.
- [49] Marco Scutari. bnlearn. <https://www.bnlearn.com/>.

- [50] Marco Scutari, Catharina Elisabeth Graafland, and José Manuel Gutiérrez. Who learns better Bayesian network structures: Accuracy and speed of structure learning algorithms. *International Journal of Approximate Reasoning*, 115:235–253, 2019. <https://arxiv.org/abs/1805.11908>.
- [51] Nitish Srivastava, G E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. <http://www.cs.toronto.edu/~rsalakhu/papers/srivastava14a.pdf>.
- [52] StatQuest. XGBoost Parts 1-4 (videos). <https://statquest.org/video-index/>.
- [53] Russell Stuart and Norvig Peter. Artificial intelligence-a modern approach 3rd ed, 2016.
- [54] Masayoshi Takahashi. Statistical inference in missing data by MCMC and non-MCMC multiple imputation algorithms: Assessing the effects of between-imputation iterations. *Data Science Journal*, 16, 2017. <https://datascience.codata.org/articles/10.5334/dsj-2017-037/>.
- [55] theinvestorsbook.com. Pert analysis. <https://theinvestorsbook.com/pert-analysis.html>.
- [56] Jin Tian and Judea Pearl. Probabilities of causation: Bounds and identification. *Annals of Mathematics and Artificial Intelligence*, 28(1):287–313, 2000. https://ftp.cs.ucla.edu/pub/stat_ser/r271-A.pdf.
- [57] Robert R. Tucci. Bayesian networks (aka causal models, DAGs) and the passage of time. blog post in blog Quantum Bayesian Networks, <https://qbnets.wordpress.com/2021/07/16/bayesian-networks-aka-causal-models-dags-and-the-passage-of-time/>.
- [58] Robert R. Tucci. Bell’s inequalities for Bayesian statisticians. blog post in blog Quantum Bayesian Networks, <https://qbnets.wordpress.com/2008/09/19/bells-inequaities-for-bayesian-statistician/>.
- [59] Robert R. Tucci. Goodness of causal fit. <https://arxiv.org/abs/2105.02172>.
- [60] Robert R. Tucci. JudeasRx. <https://github.com/rrtucci/JudeasRx>.
- [61] Robert R. Tucci. Quantum d-separation and quantum belief propagation. <https://arxiv.org/abs/2012.09635>.
- [62] Robert R. Tucci. Quantum Fog. <https://github.com/artiste-qb-net/quantum-fog>.

- [63] W. E. Vesely, F. F. Goldberg, N. H. Roberts, and D. F. Haasl. Fault tree handbook nureg-0492. <https://www.nrc.gov/reading-rm/doc-collections/nuregs/staff/sr0492/>.
- [64] Lilian Weng. The multi-armed bandit problem and its solutions. lilianweng.github.io/lil-log, <http://lilianweng.github.io/lil-log/2018/01/23/the-multi-armed-bandit-problem-and-its-solutions.html>, 2018.
- [65] Wikipedia. AdaBoost. <https://en.wikipedia.org/wiki/AdaBoost>.
- [66] Wikipedia. Autoregressive model. https://en.wikipedia.org/wiki/Autoregressive_model.
- [67] Wikipedia. Autoregressive moving-average model. https://en.wikipedia.org/wiki/Autoregressive%E2%80%93moving-average_model.
- [68] Wikipedia. Baum-Welsh algorithm. https://en.wikipedia.org/wiki/Baum%E2%80%93Welch_algorithm.
- [69] Wikipedia. Belief propagation. https://en.wikipedia.org/wiki/Belief_propagation.
- [70] Wikipedia. Berkson's paradox. https://en.wikipedia.org/wiki/Berkson%27s_paradox.
- [71] Wikipedia. Bernoulli distribution. https://en.wikipedia.org/wiki/Bernoulli_distribution.
- [72] Wikipedia. Beta distribution. https://en.wikipedia.org/wiki/Beta_distribution.
- [73] Wikipedia. Beta function. https://en.wikipedia.org/wiki/Beta_function.
- [74] Wikipedia. Binary decision diagram. https://en.wikipedia.org/wiki/Binary_decision_diagram.
- [75] Wikipedia. Boolean algebra. https://en.wikipedia.org/wiki/Boolean_algebra.
- [76] Wikipedia. Bootstrap aggregating. https://en.wikipedia.org/wiki/Bootstrap_aggregating.
- [77] Wikipedia. Categorical distribution. https://en.wikipedia.org/wiki/Categorical_distribution.
- [78] Wikipedia. Chi-square distribution. https://en.wikipedia.org/wiki/Chi-square_distribution.

- [79] Wikipedia. Chow-Liu tree. https://en.wikipedia.org/wiki/Chow%E2%80%93Liu_tree.
- [80] Wikipedia. Cochran's theorem. https://en.wikipedia.org/wiki/Cochran%27s_theorem.
- [81] Wikipedia. Conjugate prior. https://en.wikipedia.org/wiki/Conjugate_prior.
- [82] Wikipedia. Cross-validation. [https://en.wikipedia.org/wiki/Cross-validation_\(statistics\)](https://en.wikipedia.org/wiki/Cross-validation_(statistics)).
- [83] Wikipedia. Data processing inequality. https://en.wikipedia.org/wiki/Data_processing_inequality.
- [84] Wikipedia. Dirichlet distribution. https://en.wikipedia.org/wiki/Dirichlet_distribution.
- [85] Wikipedia. Errors in variables models. https://en.wikipedia.org/wiki/Errors-in-variables_models.
- [86] Wikipedia. Expectation maximization. https://en.wikipedia.org/wiki/Expectation%E2%80%93maximization_algorithm.
- [87] Wikipedia. Frisch-Waugh-Lovell theorem. https://en.wikipedia.org/wiki/Frisch%20Waugh%20Lovell_theorem.
- [88] Wikipedia. Gamma distribution. https://en.wikipedia.org/wiki/Gamma_distribution.
- [89] Wikipedia. Gamma function. https://en.wikipedia.org/wiki/Gamma_function.
- [90] Wikipedia. Gated recurrent unit. https://en.wikipedia.org/wiki/Gated_recurrent_unit.
- [91] Wikipedia. Gibbs sampling. https://en.wikipedia.org/wiki/Gibbs_sampling.
- [92] Wikipedia. Granger causality. https://en.wikipedia.org/wiki/Granger_causality.
- [93] Wikipedia. Hidden Markov model. https://en.wikipedia.org/wiki/Hidden_Markov_model.
- [94] Wikipedia. Hoeffding's inequality. https://en.wikipedia.org/wiki/Hoeffding%27s_inequality.

- [95] Wikipedia. Importance sampling. https://en.wikipedia.org/wiki/Importance_sampling.
- [96] Wikipedia. Instrumental variables estimation. https://en.wikipedia.org/wiki/Instrumental_variables_estimation.
- [97] Wikipedia. Inverse transform sampling. https://en.wikipedia.org/wiki/Inverse_transform_sampling.
- [98] Wikipedia. Jackknife resampling. [https://en.wikipedia.org/wiki/Cross-validation_\(statistics\)](https://en.wikipedia.org/wiki/Cross-validation_(statistics)).
- [99] Wikipedia. Junction tree algorithm. https://en.wikipedia.org/wiki/Junction_tree_algorithm.
- [100] Wikipedia. k-means clustering. https://en.wikipedia.org/wiki/K-means_clustering.
- [101] Wikipedia. Kalman filter. https://en.wikipedia.org/wiki/Kalman_filter.
- [102] Wikipedia. Kernel method. https://en.wikipedia.org/wiki/Kernel_method.
- [103] Wikipedia. Kernel perceptron. https://en.wikipedia.org/wiki/Kernel_perceptron.
- [104] Wikipedia. Least squares. https://en.wikipedia.org/wiki/Least_squares.
- [105] Wikipedia. Likelihood-ratio test. https://en.wikipedia.org/wiki/Likelihood-ratio_test.
- [106] Wikipedia. Linear regression. https://en.wikipedia.org/wiki/Linear_regression.
- [107] Wikipedia. Long short term memory. https://en.wikipedia.org/wiki/Long_short-term_memory.
- [108] Wikipedia. Markov blanket. https://en.wikipedia.org/wiki/Markov_blanket.
- [109] Wikipedia. Metropolis-Hastings method. https://en.wikipedia.org/wiki/Metropolis%20%93Hastings_algorithm.
- [110] Wikipedia. Minimum spanning tree. https://en.wikipedia.org/wiki/Minimum_spanning_tree.
- [111] Wikipedia. Monte Carlo methods. https://en.wikipedia.org/wiki/Category:Monte_Carlo_methods.

- [112] Wikipedia. Moving-average model. https://en.wikipedia.org/wiki/Moving-average_model.
- [113] Wikipedia. Multinomial distribution. https://en.wikipedia.org/wiki/Multinomial_distribution.
- [114] Wikipedia. Multinomial theorem. https://en.wikipedia.org/wiki/Multinomial_theorem.
- [115] Wikipedia. Multivariate normal distribution. https://en.wikipedia.org/wiki/Multivariate_normal_distribution.
- [116] Wikipedia. Natural experiment. https://en.wikipedia.org/wiki/Natural_experiment.
- [117] Wikipedia. Non-negative matrix factorization. https://en.wikipedia.org/wiki/Non-negative_matrix_factorization.
- [118] Wikipedia. Ordinary least squares. https://en.wikipedia.org/wiki/Ordinary_least_squares.
- [119] Wikipedia. Program evaluation and review technique. https://en.wikipedia.org/wiki/Program_evaluation_and_review_technique.
- [120] Wikipedia. Random forest. https://en.wikipedia.org/wiki/Random_forest.
- [121] Wikipedia. Receiver operating characteristic. https://en.wikipedia.org/wiki/Receiver_operating_characteristic.
- [122] Wikipedia. Rejection sampling. https://en.wikipedia.org/wiki/Rejection_sampling.
- [123] Wikipedia. Score test. https://en.wikipedia.org/wiki/Score_test.
- [124] Wikipedia. Simple linear regression. https://en.wikipedia.org/wiki/Simple_linear_regression.
- [125] Wikipedia. Simpson's paradox. https://en.wikipedia.org/wiki/Simpson's_paradox.
- [126] Wikipedia. Spring system. https://en.wikipedia.org/wiki/Spring_system.
- [127] Wikipedia. Student's t-distribution. https://en.wikipedia.org/wiki/Student%27s_t-distribution.
- [128] Wikipedia. Support vector machine. https://en.wikipedia.org/wiki/Support-vector_machine.

- [129] Wikipedia. Time series. https://en.wikipedia.org/wiki/Time_series.
- [130] Wikipedia. Transfer learning. https://en.wikipedia.org/wiki/Transfer_learning.
- [131] Wikipedia. Transformer (machine learning model). [https://en.wikipedia.org/wiki/Transformer_\(machine_learning_model\)](https://en.wikipedia.org/wiki/Transformer_(machine_learning_model)).
- [132] Wikipedia. Uplift modelling. https://en.wikipedia.org/wiki/Uplift_modelling.
- [133] Wikipedia. Variational Bayesian methods. https://en.wikipedia.org/wiki/Variational_Bayesian_methods.
- [134] Wikipedia. Vector autoregression. https://en.wikipedia.org/wiki/Vector_autoregression.
- [135] Wikipedia. Viterbi algorithm. https://en.wikipedia.org/wiki/Viterbi_algorithm.
- [136] Wikipedia. Wald test. https://en.wikipedia.org/wiki/Wald_test.
- [137] Hao Wu and Zhaohui Steve Qin. course notes, BIOS731: Advanced statistical computing, 2016 Emory Univ. <http://web1.sph.emory.edu/users/hwu30/teaching/statcomp/statcomp.html>.