Methodology of Educational Measurement and Assessment

Alina A. von Davier
Robert J. Mislevy
Jiangang Hao *Editors*

Computational Psychometrics: New Methodologies for a New Generation of Digital Learning and Assessment

With Examples in R and Python

Springer

Methodology of Educational Measurement and Assessment

Series Editors
Bernard Veldkamp, Research Center for Examinations and Certification (RCEC), University of Twente, Enschede, The Netherlands
Matthias von Davier, Boston College, Boston, USA

Editorial Board Members
Claus H. Carstensen, University of Bamberg, Bamberg, Germany
Hua-Hua Chang, Purdue University, West Lafayette, USA
Hong Jiao, University of Maryland, College Park, USA
David Kaplan, University of Wisconsin-Madison, Madison, USA
Jonathan Templin, The University of Iowa, Iowa City, USA
Andries van der Ark, Res Inst of Child Devt & Education, University of Amsterdam, Amsterdam, The Netherlands

This book series collates key contributions to a fast-developing field of education research. It is an international forum for theoretical and empirical studies exploring new and existing methods of collecting, analyzing, and reporting data from educational measurements and assessments. Covering a high-profile topic from multiple viewpoints, it aims to foster a broader understanding of fresh developments as innovative software tools and new concepts such as competency models and skills diagnosis continue to gain traction in educational institutions around the world. Methodology of Educational Measurement and Assessment offers readers reliable critical evaluations, reviews and comparisons of existing methodologies alongside authoritative analysis and commentary on new and emerging approaches. It will showcase empirical research on applications, examine issues such as reliability, validity, and comparability, and help keep readers up to speed on developments in statistical modeling approaches. The fully peer-reviewed publications in the series cover measurement and assessment at all levels of education and feature work by academics and education professionals from around the world. Providing an authoritative central clearing-house for research in a core sector in education, the series forms a major contribution to the international literature.

More information about this series at http://www.springer.com/series/13206

Alina A. von Davier • Robert J. Mislevy
Jiangang Hao
Editors

Computational Psychometrics: New Methodologies for a New Generation of Digital Learning and Assessment

With Examples in R and Python

Springer

Editors
Alina A. von Davier
Duolingo and EdAstra Tech, LLC
Newton, MA, USA

Robert J. Mislevy
Educational Testing Service
Princeton, NJ, USA

Jiangang Hao
Educational Testing Service
Princeton, NJ, USA

ISSN 2367-170X ISSN 2367-1718 (electronic)
Methodology of Educational Measurement and Assessment
ISBN 978-3-030-74393-2 ISBN 978-3-030-74394-9 (eBook)
https://doi.org/10.1007/978-3-030-74394-9

© Springer Nature Switzerland AG 2021
This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

The future is already here—it's just not very evenly distributed.

—attributed to W. Gibson

Digitally based learning and assessment systems generate complex data that record learners' interaction with the tasks or items at finer time granularity than in traditional settings. These rich process data can provide new opportunities for validating assessments, improving measurement precision, revealing response patterns/styles, uncovering group differences, detecting unintended behaviors, identifying new constructs, and providing more actionable feedback to learners and other stakeholders. But these benefits do not come for free. The significantly increased volume, velocity, and variety of data pose new challenges to researchers in psychometrics to handle, analyze, and interpret them in order to materialize the value of these rich data.

The techniques needed to handle these complex data are not well-covered in most existing treatments of psychometric methods. It has become necessary to extend psychometric methodology to new techniques, such as those for creating quantitative representations of the complex responses (for example, video, audio, and keystroke) and those for relating the complex evidentiary representations to the targeted constructs.

A concise new term would be beneficial to facilitate the exchanges among researchers on this new front of psychometrics. However, coming up with a concise name to encompass a discipline with such a vast intension is challenging. After years of back and forth, we noted that despite their widespread differences the new methodologies share one common feature: computational models. We thus arrived at the term Computational Psychometrics (von Davier, 2015). We want to capture the essential feature of this new discipline, encompassing data-driven computational algorithms, while at the same time, establishing its alignment to fundamental concepts of psychometrics. This may not be a perfect name, as always in the history of science, but it is a concise one that highlights the key features of this new discipline.

Computational psychometrics thus aims at integrating techniques from data science and machine learning into psychometrics, guided by well-established psychometric principles in measurement science. Just as psychometrics is not a simple collection of statistical methods for educational measurement, computational psychometrics is more than a simple aggregation of data science and machine learning methods for use in measurement contexts. Insights into educational measurement are essential for realizing the potential of applying the new techniques in those contexts.

By drawing examples from real-world use cases, this edited volume is intended to further define the scope of this new discipline and to serve as a springboard for students and researchers in psychometrics prepare for the design, development, and analysis of the learning and assessment systems with their increasingly big and complex data. A strength of the volume resides in its GitHub-site companion, which provides a repository for the code—in R or Python—for all of the methodological chapters.

This volume mirrors the societal changes, advances in technology and computational power, and the wide-spread adoption of digital learning and assessments. It is our goal that the methods provided in this book will enable researchers and developers to create systems and tools for better access, more affordability, broader inclusion, and higher quality education for everyone, everywhere.

Newton, MA, USA — Alina A. von Davier

Princeton, NJ, USA — Robert J. Mislevy

Princeton, NJ, USA — Jiangang Hao

March 14, 2021

Reference

von Davier, A. A. (2015). Virtual and collaborative assessments: Examples, implications, and challenges for educational measurement. In *Workshop on machine learning for education, international conference on machine learning*, Lille.

Contents

Contributors

Jessica Andrews-Todd Educational Testing Service, Princeton, NJ, USA

Ryan S. Baker University of Pennsylvania, Philadelphia, PA, USA

Timo Bechger Metior Consulting, Arnhem, The Netherlands

Maria Bolsinova Tilburg University, Tilburg, The Netherlands

Sebastiaan de Klerk eX:plain/University of Twente, Enschede, The Netherlands

Kristen DiCerbo Khan Academy, Mountain View, CA, USA

Michael Flor Educational Testing Service, Princeton, NJ, USA

Peter Halpin University of North Carolina at Chapel Hill, Chapel Hill, NC, USA

Jiangang Hao Educational Testing Service, Princeton, NJ, USA

Yuchi Huang ACT, Iowa City, IA, USA

Saad M. Khan FineTune Learning, New York, NY, USA

Michelle LaMar Educational Testing Service, Princeton, NJ, USA

Gunter Maris Metior Consulting, Arnhem, The Netherlands
University of Arnhem, Arnhem, The Netherlands

Maarten Marsman University of Amsterdam, Amsterdam, The Netherlands

Robert J. Mislevy Educational Testing Service, Princeton, NJ, USA

Lu Ou Unaffiliated, Campbell, CA, USA

Maria Ofelia Z. San Pedro ACT, Inc., Iowa City, IA, USA

Josine Verhagen McGraw Hill Education, New York, NY, USA

Alina A. von Davier Duolingo and EdAstra Tech, LLC, Newton, MA, USA

Pak Chung Wong University of Iowa, Iowa City, IA, USA

Mengxiao Zhu University of Science and Technology of China, Hefei, Anhui, China

Chapter 1
Introduction to Computational Psychometrics: Towards a Principled Integration of Data Science and Machine Learning Techniques into Psychometrics

Alina A. von Davier, Robert J. Mislevy, and Jiangang Hao

Abstract In this chapter we articulate what is computational psychometrics, why we need a volume focused on it, and how this book contributes to the expansion of psychometric toolbox to include methodologies from machine learning and data science in order to address the complexities of big data collected from virtual learning and assessment systems. We also discuss here the structure of the edited volume, how each chapter contributes to enhancing the psychometrics science and our recommendations for further readings.

Psychometrics originated more than a century ago as the application of quantitative methods to support inferences about psychological attributes and processes from observable data. In broad terms, it has remained thus, even as a great deal has changed. Statistical theory, models, and methods have progressed. Cognitive, sociocultural, situative, and neurological research has remade psychology. Advances in technology have spurred sophistication in measurement instruments.

And, most recently and most dramatically, the quantity, varieties, transportability, and ubiquity of *data* itself have exploded (Behrens & DiCerbo, 2014). It is now feasible to capture keystrokes and pathways through interactive digital investigations, and log traces of solution processes to traditional and innovative tasks. Data can be

The R or Python codes can be found at the GitHub repository of this book: https://github.com/jgbrainstorm/computational_psychometrics

A. A. von Davier (✉)
Duolingo and EdAstra Tech, LLC, Newton, MA, USA
e-mail: avondavier@duolingo.com

R. J. Mislevy · J. Hao
Educational Testing Service, Princeton, NJ, USA
e-mail: rmislevy@umd.edu; jhao@ets.org

© Springer Nature Switzerland AG 2021

A. A. von Davier et al. (eds.), *Computational Psychometrics: New Methodologies for a New Generation of Digital Learning and Assessment*, Methodology of Educational Measurement and Assessment, https://doi.org/10.1007/978-3-030-74394-9_1

marshalled from students' actions to provide feedback and adapt challenges in real time, and data can be brought together across sources within and beyond instruction.

We, as editors and authors, intend in this volume to contribute to the expansion of psychometric science principally as to the last of these changes, the burgeoning of data, in an enterprise we are calling *computational psychometrics* for short. Alina von Davier (2015) introduced the term in an invited talk at the International Conference of Machine Learning at Lille, France, in July of 2015, and subsequently defined it more formally as a blend of data science techniques, computer science, and principled psychometric approaches to aid the analyses of complex data as obtained from performance assessments and technology-enhanced learning and assessment systems (LAS) (von Davier, 2017).

Much work has already appeared along these lines, of course. Interdisciplinary researchers have developed a variety of innovative methods for data wrangling, analytics, modeling, and prediction of students' success in all manner of digital learning and assessment systems. Contributions to the research literature appear under appellations such as network psychometrics (Epskamp et al., 2018), educational data mining (Baker, 2010), intensive longitudinal data models (Hamaker & Wichers, 2017), and machine learning in education (von Davier et al., 2016).

It is not easy to encapsulate this emerging work as it arises along multiple fronts to meet multiple challenges. Despite the multiplicity of new methodologies, we note that they share a common feature, intensive computation, and therefore adopted the term "computational psychometrics". Most of this work is quite technical and specialized—less known to students, working professionals, or experts in some of the areas but new to other areas. For these readers, we hope this volume can provide a foundation for integrating advances in data science and machine learning; this integration enhances applications that we can describe as psychometrics in the broad sense, by extending fundamental concepts and principles that have been developed in psychometrics over the decades.

The chapters in this book work through the integration we propose mainly with applications in education, particularly in assessment, although the ideas and the methodologies apply more broadly. Not only does assessment happen to be the field in which we have worked most extensively, it is an area of great importance and in great need. The most familiar psychometric methods in assessment have been developed for high-stakes testing, with relatively sparse and simple data forms, for uses such as selection and summative evaluation that lie outside the immediate context of learning activities. Current interests in assessment concern integration with learning; distribution over time and place; synthesis with other sources of information; and virtual performance assessments (VPAs) that offer options for interactivity, richer and multimodal data, and collaboration among learners. It is not simply that there are more data; the data differ qualitatively. They can reveal new aspects of thinking and acting more directly and at a finer grain size, although presenting complexities such as interdependencies, capabilities that change over time, and observational contexts that evolve over time and vary across students.

Computational psychometrics is more than a simple aggregation of methods. We emphasize that data collected from digital environments should be intentional to the

extent possible. The data collection design expands the widely accepted evidence-centered design framework for educational assessments (ECD; Mislevy et al., 2003; also see Mislevy, 2018) for designing environments and providing opportunities for students to learn and to display the skills we want to develop and assess. The ECD framework further ensures that the identification of evidence is rooted in the science of learning (Arieli-Attali et al., 2019). Computational psychometrics both draws on and furthers disciplinary theory as a map for measurement efforts, and leverages any combination of direct response process data (e.g., response time, persistence, procrastination, compliance with instructions, metacognition and problem-solving strategies) and other ancillary data (e.g., demographics, geospatial, temporal) to shed light on what people know and can do, and what learning paths they might most benefit from given their current mastery state and learning objectives.

A dedicated research center, initially under the name of Center for Advanced Psychometrics, later renamed as the Computational Psychometrics Research Center (CPRC), was founded at Educational Testing Service in 2013 to promote research in this area. The editors originated this volume at CPRC during this time. In 2016, ACT, Inc. established the ACTNext division under Alina von Davier, where R&D-based innovations were situated within the computational psychometrics framework (von Davier et al., 2019a, b). Many of the authors of the following chapters have been associated with these groups and the synergies their collaborations have produced. The ACTNext EdTech and Computational Psychometrics Symposium organized annually in the Iowa City area between 2016—2019[1] brought together researchers and EdTech developers to explore the synergies across disciplines. On December 3–4, 2020, the Association of Test Publishers (ATP) organized its first international EdTech and Computational Psychometrics Summit.[2] A Computational Psychometrics Wikipedia page was created in 2018.[3] Numerous invited sessions, workshops, and tutorials on computational psychometrics were held at National Council of Measurement in Education (NCME), Educational Data Mining (EDM), International Conference of Machine Learning (ICML) and Computational Social Science over the past years. This volume illustrates the progress made in understanding and optimizing the blend of psychometric theory and data-driven algorithms.

Contents

The contributors to this volume illustrate how the interdisciplinary chapters and their particular areas of expertise blend into a coherent approach to handle the increasingly complex data collected from digital assessments and integrated learning-and-assessment systems. We focus on measurement-related aspects of the process of building such assessments, and discuss types of data collected from them and the dependencies in the data that force psychometricians to rethink their approaches in

[1] http://etcps.actnext.info/2019/

[2] http://www.globalatpevents.com/ecps/

[3] https://en.wikipedia.org/wiki/Computational_psychometrics#:~:text=Computational%20Psychometrics%20is%20an%20interdisciplinary,%2C%20biometric%2C%20or%20psychological%20data

order to measure complex constructs. We lay out specific methodologies that have been found useful to analyze data from learners' processes in interacting with and responding to virtual environments, illustrating the ideas with real data examples and corresponding computer code.

The volume is organized in two parts. From one perspective, the work in computational psychometrics we describe extends hard-won psychometric insights to a larger universe of constructs, data types, and technological environments. Part I thus offers an overview of the changes in the educational landscape, with a focus on the transition to the next generation of learning and assessment and an overview of selected areas of technological development that have the potential to transform the educational experience. From another perspective, the work we describe provides foundations for harnessing the power of advanced data analytics and machine learning methods to the particular problems of assessment. Part II is focused on specific methodologies that are relevant illustrations of computational psychometrics.

More specifically, Part I brings out the connection between enduring evidentiary and ethical groundings and the newly emerging computational and analytic methods. Although there are many data-driven (i.e., theory-neutral) methods currently available for exploring and analyzing "big data," few experts have integrated these methods with insights that have been developed over decades of research and experience in psychometrics and educational measurement. Theory and methods have evolved for addressing fundamental principles of reliability, validity, comparability, generalizability, and fairness, instantiated originally with simpler forms of data. Yet, as Samuel Messick (1994) put it, these are not just measurement issues, but "*social values* that have meaning and force outside of measurement wherever evaluative judgments and decisions are made" (p. 16; emphasis original).

Part I begins with Chap. 2, in which Robert Mislevy provides a brief look at foundations in sociocognitive psychology and assessment theory that computational psychometrics requires. In Chap. 3, Alina von Davier, Kristen DiCerbo, and Josine Verhagen define computational psychometrics as a framework in the virtual environment of learning and assessment, such as games, simulations, collaborations, and augmented environments, which often result in rich data of complex structure. In Chap. 4, Jessica Andrews-Todd, Robert Mislevy, Michelle LaMar, and Sebastiaan de Klerk describe design and methodology for virtual performance assessments. In Chap. 5, Sweet San Pedro and Ryan Baker provide an overview of adaptive learning and of data mining techniques used to identify learners' level and the timing of when they are ready to move to the next level.

Part II focuses on specific methodologies that are relevant illustrations of computational psychometrics. Developing innovative assessments is in high demand domestically and internationally. R and python (open source software) are currently popular for statistical applications and multimodal big data analytics respectively. This set of chapters contains substantive introductions to advanced analytic concepts for uses as a reference and as a handbook. Illustrative code for R and python will aid readers in developing new applications. They will also help students majoring

in educational measurement to equip themselves with necessary analysis techniques for their future career.

Part II begins with Chap. 6, in which Robert Mislevy and Maria Bolsinova cover traditional psychometrics with an emphasis on the extension of foundational ideas to computational psychometrics. In the next chapter, Chap. 7, Gunter Maris, Timo Bechger, and Maarten Marsman provide an introduction to Bayesian inference using Markov Chain Monte Carlo (MCMC) methods. They illustrate how a wide range of computational models can be estimated within the MCMC framework.

In Chap. 8, Jiangang Hao and Robert Mislevy give an overview of data science and introduce some data techniques and strategies for process data from digitally based assessments. The next three chapters, from 9 to 11, provide a gentle introduction to machine learning and AI, with example applications in educational context. We do not intend to provide a comprehensive review of machine learning methods through the three short chapters, but rather hope they serve as introductions for educational researchers (especially those with a traditional psychometrics backgrounds) to the realm of machine learning. Jiangang Hao introduces supervised machine learning in Chap. 9 and Pak Chung Wong discusses unsupervised methodologies and their applications in Chap. 10. Yuchi Huang and Saad Khan's Chap. 11 discusses advances of Artificial Intelligence (AI) techniques, such as Deep Neural Network (DNN) and Convolutional Neural Network (CNN), and their roles in analyzing multimodal data, with an example for measuring the distance between images. Peter Halpin, Ou Lu, and Michelle LaMar's Chap. 12 focuses on time series and stochastic processes methodologies, and illustrates their application for modelling interactive learning and assessment. Mengxiao Zhu's Chap. 13 focuses on social network analysis (SNA) and its potential in visualizing and characterizing process data for learning and assessments. The final chapter, Chap. 14, by Michael Flor and Jiangang Hao introduces some basics of natural language processing (NLP) and their typical applications in educational contexts, such as text mining and automated scoring.

The computer codes for the examples that appear in these chapters are available in a repository on GitHub at: https://github.com/jgbrainstorm/computational_psychometrics. The link also appear in the footnotes for each chapter that includes programming code, so that the chapters can stand independently as supplementary material for focused purposes in instruction or applications.

Audience The book is intended for undergraduate and graduate students, faculty, researchers, practitioners at testing institutions, and anyone in psychometrics, measurement, education, psychology, and other fields who is interested in adaptive learning and testing, intelligent tutoring systems, or game- and simulation-based assessment and learning, especially with respect to practical implementations of algorithms using R and python. It is both an entry point and a bridge for people with traditional psychometric backgrounds to come up to date with techniques for handling more complex data from digital learning and assessment systems. The book is also intended to be useful for professionals and practitioners interested in learning about emerging computational solutions.

A volume such as this is but a snap-shot in time. Given the speed at which the technology and methodology that support machine learning (ML) algorithms progresses, the specific code we offer may be less relevant in a few years. However, we have designed the book in accordance to first principles—theory, statistical considerations, technology design—and therefore we hope that it will prove useful to practitioners and researchers in psychometrics and educational technology for years to come.

References

Arieli Attali, M., Ward, S., Thomas, J., Deonovic, B., & von Davier, A. A. (2019). The expanded Evidence-Centered Design (e-ECD) for learning and assessment systems: A framework for incorporating learning goals and process within assessment design. *Frontiers in Psychology*. https://www.frontiersin.org/articles/10.3389/fpsyg.2019.00853/full

Baker, R. S. J. D. (2010). Data mining for education. In B. McGaw, P. Peterson, & E. Baker (Eds.), *International encyclopedia of education* (Vol. 7, 3rd ed., pp. 112–118). Elsevier.

Behrens, J. T., & DiCerbo, K. E. (2014). Harnessing the currents of the digital ocean. In J. Larusson & B. White (Eds.), *Learning analytics* (pp. 39–60). Springer.

Hamaker, E. L., & Wichers, M. (2017). No time like the present: Discovering the hidden dynamics in intensive longitudinal data. *Current Directions in Psychological Science, 26*, 10–15.

Epskamp, S., Maris, G., Waldorp, L. J., & Borsboom, D. (2018). Network psychometrics. In P. Irwin, T. Booth, & D. J. Hughes (Eds.), *The Wiley handbook of psychometric testing: A multidisciplinary reference on survey, scale and test development*. Wiley Online Library. https://onlinelibrary.wiley.com/doi/book/10.1002/9781118489772

Hao, J., & Ho, T. K. (2019). Machine learning made easy: A review of Scikit-learn package in Python programming language. *Journal of Educational and Behavioral Statistics, 44*(3), 348–361.

Mislevy, R. J. (2018). *Sociocogntive foundations of educational assessment*. Routledge.

Mislevy, R. J., Steinberg, L. S., & Almond, R. A. (2003). On the structure of educational assessments. *Measurement: Interdisciplinary Research and Perspectives, 1*, 3–67.

Messick, S. (1994). The interplay of evidence and consequences in the validation of performance assessments. *Educational Researcher, 23*(2), 13–23.

von Davier, A. A. (2015). *Computational psychometrics*. Invited presentation at the pre-conference workshop "Machine Learning in Education" at the International Conference of Machine Learning, Lille, France.

von Davier, A. A. (2017). Computational psychometrics in support of collaborative assessments. In A.A. von Davier (Ed.), Measurement issues in collaborative learning and assessment (Special Issue). *Journal of Educational Measurement, 54*(1), 3–11.

von Davier, A. A., Chung Wong, P., Yudelson, M., & Polyak, S. (2019a). The argument for a "data cube" for large-scale psychometric data. *Frontiers in Education*. https://doi.org/10.3389/feduc.2019.00071

von Davier, A. A., Deonovic, B., Yudelson, M., Polyak, S., & Woo, A. (2019b). Computational psychometrics approach for holistic learning and assessment systems. *Frontiers in Education*. https://www.frontiersin.org/articles/10.3389/feduc.2019.00069/full

von Davier, A. A., van der Schaar, M., & Baraniuk, R. (2016). Machine learning for digital education and assessment Systems. A pre-conference workshop, International Conference of Machine Learning. https://icml.cc/2016/index.html%3Fp=1519.html. https://www.frontiersin.org/articles/10.3389/feduc.2019.00071/full

Part I
Conceptualization

Chapter 2
Next Generation Learning and Assessment: What, Why and How

Robert J. Mislevy

Abstract Computational psychometrics is a blend of stochastic processes theory, computer science-based methods, and theory-based psychometric approaches that may aid the analyses of complex data from performance assessments. This chapter discusses the grounds for using complex performance assessments, the design of such assessments so that useful evidence about targeted abilities will be present in the data to be analysed, and roles that computational psychometric ideas and methods can play. It first provides background on a situative, sociocognitive, perspective on human capabilities and how we develop them and use them—a perspective we believe is necessary to synthesize the methodologies. Next it reviews the form of evidentiary argument that underlies the evidence-centered approach to design, interpretation, and use of educational assessments. It then points out junctures in extensions of the argument form where computational psychometric methods can carry out vital roles in assessment of more advanced constructs, from more complex data, in new forms and contexts of assessment. It concludes by reflecting on how one reconceives and extends the notions of validity, reliability, comparability, fairness, and generalizability to more complex assessments and analytic methods.

2.1 Introduction

Von Davier (2017) describes computational psychometrics as "a blend of stochastic processes theory, computer science-based methods, and theory-based psychometric

The R or Python codes can be found at the GitHub repository of this book: https://github.com/jgbrainstorm/computational_psychometrics

R. J. Mislevy (✉)
Educational Testing Service, Princeton, NJ, USA
e-mail: rmislevy@umd.edu

© Springer Nature Switzerland AG 2021

A. A. von Davier et al. (eds.), *Computational Psychometrics: New Methodologies for a New Generation of Digital Learning and Assessment*, Methodology of Educational Measurement and Assessment, https://doi.org/10.1007/978-3-030-74394-9_2

approaches that may aid the analyses of complex data from performance assessments" (p. 4). Most of the chapters in this volume address the "how" of *applying* computational psychometric methods with complex data from performance assessments and related venues such as learning systems and online courses. I back up two steps in this chapter, to the "why" of *using* complex performance assessments in the first place, and the "how" of *designing* such assessments so that useful evidence about targeted abilities will be present in the data to be analysed.

To these ends, I review in Sect. 2.2 a situative, sociocognitive, perspective on the nature of human capabilities, how they are developed, and how we employ them to interact with the social and physical world (Greeno, 1998). From this perspective arises the need for a broader range of assessment types, usually involving interaction with a responding environment, often in contexts that are digital, and sometimes requiring collaboration among individuals. It is in such environments, and from test-takers' actions in them, that direct evidence of capabilities involving inquiry, problem-solving, communication, and the like can be elicited.

In Sect. 2.3 I describe an evidentiary-argument framework that helps developers work from initial conceptions of targeted capabilities to environments, affordances, and activities that can produce evidence. The discussion draws on evidence-centered design (Mislevy et al., 2003) as it applies to game- and simulation-based assessments (Mislevy, 2018; Mislevy et al., 2012). This section notes locations and roles for computational psychometric methods.

Developments in technology and learning science have put educational assessment on the threshold of a paradigmatic advance. Computational psychometric methods like the ones described in this volume are indispensable. Much has changed already, with regard to data, models, analytic methods, and contexts of use; further change is certain. Section 2.4 addresses an aspect of assessment that should not change. Messick (1994) reminds us that "validity, reliability, comparability, and fairness are not just measurement issues, but *social values* that have meaning and force outside of measurement wherever evaluative judgments and decisions are made" (p. 13, emphasis original). The section reflects on ways that that these qualities, along with generalizability, extend to more complex assessments and analytic methods.

2.2 A Sociocognitive Psychological Perspective on Assessment

The analytic models and data-gathering methods of educational assessment originated under the perspectives of trait and behavioural psychology. Rich performance tasks were employed, as evidenced by Ryans and Frederiksen's (1951) mid-century chapter on performance assessment. But administration costs were high, and evaluation was limited to either the qualities of final products, as with essays, or the judgments of human raters, as with oral proficiency language interviews.

Accordingly, large-scale testing was limited mainly to timed, sequestered, administrations of objectively-scored tasks such as multiple-choice items, to test-takers who responded independently. Broad ranges of content domains could be assessed efficiently in this manner, albeit with the constricted range of situations, performances, and resulting data. The details of moment-by-moment activity were lost—ignored when only final products were addressed, and, when complex performances were rated by humans, evaluated with through rubrics by processes hidden in raters' heads. In both cases, the resulting data, the grist of psychometric analyses, were relatively simple: vectors of 0/1 item scores or ratings on a few ordered scales. These data and the ensuing estimation of a small number of trait values or content-domain scores were viewed as sufficient for test-score interpretations and uses cast in trait and behavioural perspectives.

Greeno et al. (1997) argued that a more encompassing psychological perspective accommodates the trait and behavioral psychological perspectives, as well as the information-processing cognitive perspective, as more specialized cases. Broadly speaking, it can be called a situative, sociocognitive, perspective (which I will call sociocognitive for short). It is important to computational psychometrics because it provides grounding for designing and analysing assessments that are interactive, in which evidence arises from moment-to-moment activity, and the capabilities of interest are not easily specified in trait or behavioral terms.

2.2.1 The Sociocognitive Perspective in a Nutshell

A situative, sociocognitive perspective shifts awareness from solely individual behavior and cognition to the larger systems in which it occurs, that of cognitive agents interacting with each other and with other subsystems in the environment. "Situative" highlights a central focus: "persons-acting-in-context"—not just final products; not just traits or behavioral tendencies of persons; and not just tasks. The "socio-" in "sociocognitive" highlights the regularities across people and across situations within which individuals act and interact (Wertsch, 1994). I call these linguistic, cultural, and substantive patterns, or *LCS patterns* for short, and recurring clusters of them *practices*. Although they vary over place and time as the social milieu varies and changes, it is these regularities across these unique events that make between-person interactions meaningful, indeed even possible (Sperber, 1996). Of particular interest in educational assessment are LCS patterns and practices that are connected with school and work, such as disciplinary ideas like energy transfer and practices like model-based reasoning in science. The "-cognitive" in "sociocognitive" highlights the attunements that individuals develop for recognizing instances of the many LCS patterns around which every situation is structured, and the resources they develop for acting effectively in such situations, as they solve problems, collaborate with others, or communicate in person or across time and space.

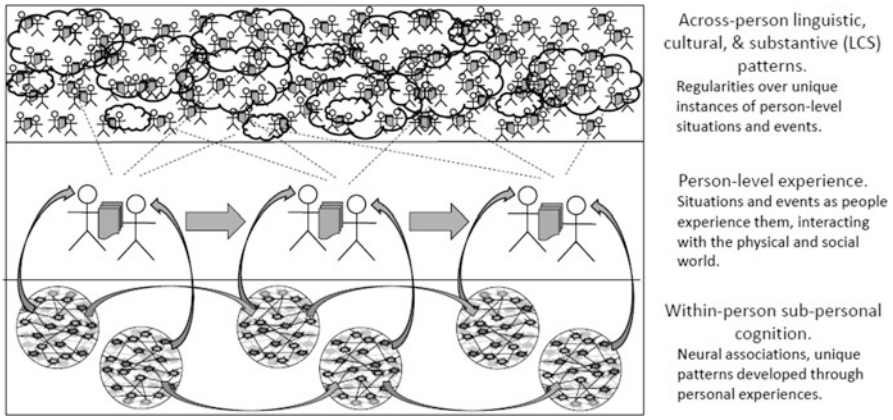


Fig. 2.1 A sociocognitive perspective entails a complex adaptive system

Individuals' cognitive resources are traces of an individual's past experiences, continually assembled, adapted, and revised to make meaning and guide action in each new situation we encounter or create. We assemble them in the moment to make our way through the physical and social world, extending and adapting them as we do. The world as we experience it is an ongoing blend of the particulars of the situation at hand and of patterns we have developed resources for, through our previous experiences structured around LCS patterns (Kintsch, 1998). Each such episode further revises and extends our cognitive resources, sometimes minimally, other times quite radically.

These and related phenomena constitute a complex adaptive system (CAS; Holland, 2006). Figure 2.1 suggests three layers that are important for understanding assessment and roles of computational psychometrics in assessment.

The layer in the middle of the figure is "person-level experience," or situations and events as we experience them, as we interact with the physical and social world. Reading a map, ordering a meal at a restaurant, and working through an assessment task are examples. The three successive replications represent the unfolding interaction, a situation that changes moment by moment as they interact. *Computational psychometrics can play a role in characterizing the nature of moment-by-moment interactions in valued activities, such as sequences of actions in scientific inquiry or conversational moves in collaboration* (Chaps. 4, 8, 11, 12 and 13).

For such an interaction to be meaningful, more must be going on at two other layers, across people and within people. The clouds in the upper layer represents the regularities, despite the uniqueness of each event, across many such person-level episodes. These are the LCS patterns. Writing about language in speech communities, for example, Gumperz (1982) noted that the variation among individuals and situations "seems irregular when observed at the level of the individual, nonetheless shows systematic regularities at the statistical level of

social facts" (p. 24). Some LCS patterns, like ones involved in science inquiry and mixed number subtraction, are the subject of explicit instruction in school and work, and therefore the subject of assessment as well. For example, one might use natural language processing (NLP) tools to characterize the ways that people use language in a domain, to provide automated feedback to students to improve their writing. *Computational psychometrics can play a role in characterizing patterns of stability and variability in social practices that hold value in a community* (Chaps. 4, 11 and 14).

The existence of such regularities across people isn't sufficient. An individual must be able to recognize LCS patterns implicit in a situation, blend them with its particulars, know how to act in it, and create successive situations. She must have developed relevant cognitive resources through her personal history of experience in situations that were also built around particular mixes of various LCS patterns. These cognitive resources are depicted in the bottom layer of the figure as associations in the literal neural network of a human brain. There must be enough similarity of resources across individuals, developed through their unique personal histories, for them to interactively build a shared interpersonal meaning concerning a given situation and the paths of action that are available. *Computational psychometrics can play a role in characterizing patterns of stability and variability in individuals' actions in certain kinds of situations, for certain purposes, for kinds of purposes* (Chaps. 4, 12, 13 and 14).

2.2.2 Implications for Learning and Assessment

A number of findings from situative and sociocognitive perspectives hold implications for learning and assessment, and for roles for computational psychometrics in turn.

LCS patterns of many kinds and many levels of granularity are involved in every situation. They are mixed and matched, used and re-used in different kinds of situations, although with variations in their regularities. All this plays out moment by moment, in person-level activities. Actions are the epiphenomena of the within-person cognitive activity depicted in the lower layer of Fig. 2.1, manifest in shared activity in the middle layer, and structured around the shared LCS patterns at the top layer.

Correspondingly, people assemble resources of many kinds and at many levels of granularity to comprehend and to act in every situation. The resources we assemble to comprehend and act in a situation are activated by the features of a situation as we recognize it (Kintsch, 1998). Physics problems that seem unrelated to a novice might all be viewed by an expert as instances of the same theme, such as equilibrium, which in turn activates resources for models and strategies to begin stepping through solutions (Ericsson et al., 2018).

The resources we develop are initially tied closely to the circumstances of learning. This is why, for example, many students can solve physics problems at the

end of the chapter but do not apply the same principles to reasoning about everyday situations. Being good at solving problems in the video game Halo does not mean a student will be good at solving problems working in a stock room or in interpersonal relationships.

Declarative knowledge is necessary for acting effectively in situations, but it is not sufficient. Experts in a domain have more knowledge than novices in a domain, but their knowledge is organized around underlying principles and with resources for effective recognition and action in relevant situations.

Both instructional experiences and assessments are particular kinds of practices, which overlap the features of practices in the real-world we want students to learn to act in. An effective way to support students' development is to provide learning experiences in facsimiles of those situations with those features and purposes, usually starting with simpler versions of them and providing feedback. This is why *The Next Generation Science Standards* (NGSS Lead States, 2013) argues for tasks that integrate disciplinary content with science practices such as constructing explanations and cross-cutting themes such as cause-and-effect.

In assessment, the most direct evidence about integrated capabilities is obtained when tasks more closely match the features, the contexts, and the purposes of the targeted capabilities. A successful outcome may point to the deployment of appropriate knowledge and activity structures, but the processing that produced the work remains hidden. In order to examine in greater detail the nature of the resources a student has employed, we need the evidence that is found in moment by moment actions in interactive situations.

Assessment from a sociocognitive perspective centers on the resources individuals have developed to act through LCS patterns, as they are organized in valued practices. Interest lies in the knowledge and activity patterns, and the pragmatic understandings of how to use them in real-world situations. Do individuals recognize markers of targeted patterns? Do they construct internal meanings in their light? Are their actions appropriate to targeted patterns and practices, and effective? What are the ranges and circumstances of activation of the resources they have developed? What kinds of performances do they tend to produce in certain domains of situations, and how might their capabilities vary across contexts?

Of course such questions have always been at the bottom of educational assessment, although cast in trait, behavioral, and more recently, information-processing terms. Framing them first in sociocognitive terms connects us with psychological first principles for design and analysis. In particular, the methods of computational psychometrics will help us frame questions and analyze observations to answer them in new forms of assessment.

2.3 An Evidentiary Reasoning Framework for Assessment Design

Assessments of all kinds can be viewed as arguments from particular observations in particular circumstances, to inferences at some higher level for purposes that lie beyond the assessment situation itself. This section reviews a basic structure for assessment design and use arguments (Kane, 1992; Mislevy et al., 2003), highlighting extensions to interactive and collaborative situations (Behrens et al., 2012). Locations where computational psychometric methodologies can be leveraged are noted. The section that then follows will use it to discuss measurement issues as social values.

Figure 2.2 is the basic argument form. It serves as a design argument prospectively, as task environments, affordances, work products, and evaluation procedures are being crafted, and as an interpretation argument retrospectively, once an examinee's enacted performance and necessary additional information are in hand. I will first walk through its elements with a familiar multiple-choice item, then extend the ideas to multiple items, interactive tasks, and collaborative work.

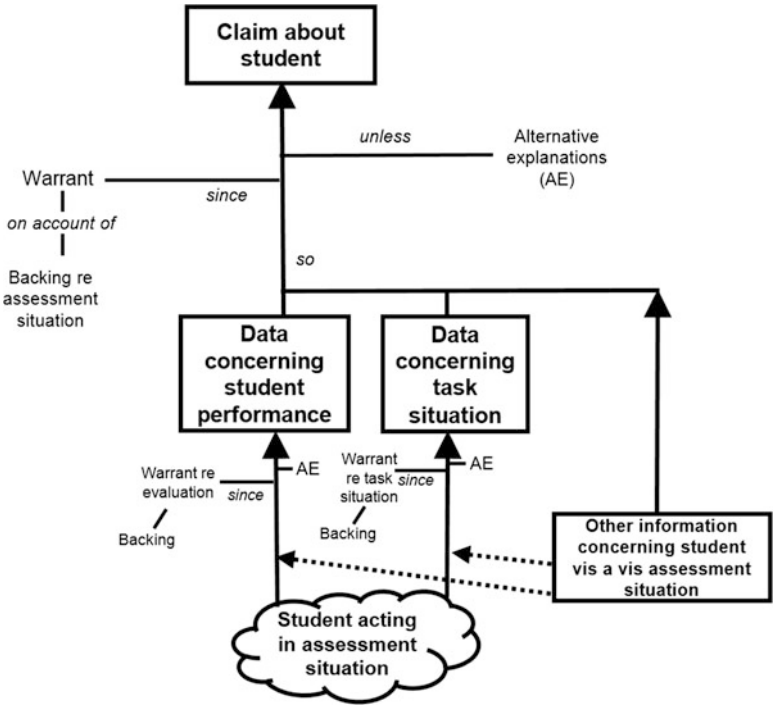


Fig. 2.2 An assessment design/interpretation argument

The cloud at the bottom of the figure represents the student's performance—what a student says, does, or makes in the assessment situation. This is the raw material for inference about students more broadly. Some applications of computational psychometrics work from patterns in data obtained in situ, such as data from corpora of texts or spontaneous student-generated chats in a Massive open online course, or MOOC. Others work from data in designed environments, such as games and simulations that have been structured to a greater degree in order to increase learning opportunities or evoke evidence about students' capabilities (Chaps. 3, 4 and 5).

At the top of the figure is a claim we wish to make: some statement about the student's capabilities, perhaps probabilistic, that holds meaning beyond simply describing a realized performance. It may be qualitative, quantitative, or a mix; it may be cast in terms drawn directly from the sociocognitive perspective or approximated more parsimoniously in terms of traits or of behavioral tendencies in some domain of situations. In many psychometric models, the claim is expressed in terms of values of latent variables associated with students, which represent traits, states, behavioral tendencies, or other psychological characteristics. The models used with traditional assessment data employ relatively few variables for student capabilities, and their values are considered to be constant during the assessment. More recent developments include dynamic models such as the Bayesian model tracing approach used in intelligent tutoring systems to track learning as students work through problems (Desmarais & Baker, 2012); finer-grained, faster-changing local variables within problems and less malleable higher-level variables across problems (Conati et al., 2002); and Markov-process models with lower-level variables to address changing states of knowledge as interactions proceed, governed by higher-level fixed parameters. These analysts assemble model fragments at the grainsizes and for the targeted aspects of capability and for the actions that arise as a student acts in the environment. *Computational psychometric modeling can be used to model students' capabilities more flexibly as both environments and capabilities themselves change in varying ways and at varying rates* (Chaps. 3, 4, and 13).

A performance might have the potential to provide information to support a claim, but this *activity* is not yet *data*. Between a performance and a claim can lie a great deal of reasoning, computation, justification, backing, and qualification. There are actually three distinct kinds of data in an assessment argument (and multiple steps to identify and evaluate them): the data concerning the performance, data concerning the situation in which it occurs, and additional information concerning the person which may be relevant to the context and the intended inference (metadata and paradata). Before describing them, however, we must consider *warrants*, the "glue" that connects data to claims (Schum, 1994), and their *backing*.

The warrant attached to the arrow going up from the data to the claim in Fig. 2.2 is the justification for reasoning from particular data to a particular claim. In assessment arguments cast in a sociocognitive perspective, the warrants are grounded ultimately in a persons' resources for acting in kinds of ways in kinds of situations. For instance, a warrant in an interactive inquiry assessment would be something like "If a student can plan a plant respiration experiment to determine the relationship among variables involving Energy and Matter, then he is more likely

to carry out a sequence of trials that are consistent with a controlling-for-variables strategy." The degree to which such evidence generalizes to other content areas and aspects of inquiry, to ground a more trait-like inference, is an empirical matter (Sect. 2.4.4).

The *backing* for a warrant is the body of theory, research, experience, and knowledge about the present situation that support the intended inference. *A role for computational psychometrics is in characterizing the patterns of knowledge and/or activity in domains or in particular kinds of situations*—that is, patterns of stability and variation across persons, which suggest spaces within which claims about individuals and actions might be expressed (Chaps. 8, 9, 10, 11 and 13).

Inference in assessment is fallible. Conditions in the warrant might not hold, and even when they do, the expected observation need not occur because the relationship is probabilistic. These *alternative explanations* weaken the claim. The more there are and the more likely they are to hold, the more vulnerable is the argument, and the weaker is the validity of the inference. Test developers try to reduce their force by good design strategies, definition of testing populations, and practice materials. Test users reduce their force when they can by incorporating additional knowledge about students into their inferences. *A caution for computational psychometrics is that patterns of stability and variation arise from particular people acting in particular times and places* (Chap. 6). Inferential errors, possibly invalid or unfair decisions, can result by presuming the same patterns hold for different populations, times, or motivations.

Now, regarding data: The first kind of data concerns the performance. The data concerning the performance are not the performances themselves, but interpretations of it, discerned from the many aspects of activity one might identify. They depend on our conception of what we think is important, how we want to use it, and the constraints we face. As with the main assessment argument, the identification and evaluation of aspects of a performance are themselves obtained through an argument–an evidence-evaluation subargument. In familiar assessment practices, the data going into psychometric models are fairly simple vectors, identified directly from features of performance by direct means (correctness of response choice, whether an engine starts after repair) or complex but hidden processes by human raters.

The data concerning the performance go forward in conjunction with the second kind of data, data concerning the situation, such as the multiple choice item or an engine with a given fault. The features of the task environment center on the LCS patterns and practices that are at issue, the actions that students can take, and the objective of the activity.

Things become more complicated when performance is interactive. Consider a single examinee interacting with a simulation. As she takes an action, the situation changes, and the meaning of subsequent actions depends on the features of the new situation. This can occur repeatedly, and proceed in different directions for different examinees. Fig. 2.3 chains arguments like those of Fig. 2.2 in sequence, showing the dependence of the situation, hence the argument, from one state to the next. *Roles for computational psychometrics can thus include identifying*

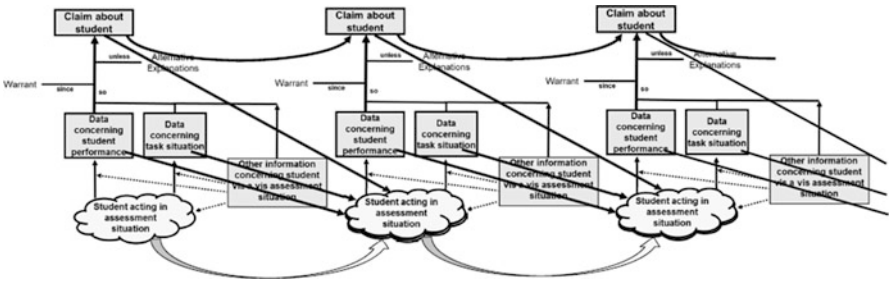


Fig. 2.3 An evolving assessment design/interpretation argument

and evaluating the relevant features of evolving situations, and *identifying and evaluating relevant features of examinee performance conditional on features of the present and possibly preceding and subsequent states of the situation* (Chaps. 9, 10, 11, 12, 13 and 14).

Procedures for identifying and evaluating performance features and situation features can be hierarchical, involving multiple stages and multiple methods. This is due to the layering of LCS patterns that constitute situations and of resources that produce comprehension and action. C*omputational psychometric methods of different kinds and at multiple stages can be employed in evidence-evaluation of performances and of features of situations* (Chaps. 8, 9, 10, 11, 12, 13 and 14). Each stage in a hierarchical evidence-identification procedure embodies its own argument, requiring a warrant and backing, and subject to alternative explanations.

Additional challenges arise when different aspects of performance are captured in multiple modalities, such as vocal utterances, facial expressions, and physical gestures, and must be integrated (Khan, 2017). *Multimodal computational psychometric methods synthesize information across different forms of data, about different aspects of performance, in terms of evidence about examinees' semantic, strategic, or cognitive actions* (Chap. 11). Psychometric modelling can be carried out at multiple levels when examinees interact. *One role that computational psychometric methods can play is modeling actions, states, strategies, and/or capabilities of* groups *as they collaborate (or compete)* (Chaps. 3 and 4). *Another role is modeling the capabilities of* individuals*, where features of the situations include the actions of other ensemble members* (Chaps. 3 and 4).

The third kind of data in assessment arguments is additional information about the examinee that is relevant to interpreting performance. An explicit example with multiple-choice data is using different item parameters for certain test items and subpopulations. An implicit example is including student background characteristics for instructional feedback, when test performance is joined with language background. *A role for computational psychometric methods is in integrating performance data with examinee information for inference* (Chaps. 6 and 7).

2.4 Not Just Measurement Issues, but Social Values

Validity, reliability, comparability, generalizability, and fairness are core values in educational assessment. Much thought has been devoted over the past century to explore their foundations, develop tools for practical work, and establish practices to ensure these values as they apply to familiar forms and uses of data. This section gives the gist of each concept and suggests how it might be extended to richer data and more complex analytic procedures.

2.4.1 Validity

The classical definition of validity is whether an assessment "measures what it is purported to measure" (Kelley, 1927, p. 14). Messick's (1989) definition[1] points the way toward practice: "Validity is an integrated evaluative judgment of the degree to which empirical evidence and theoretical rationales support the adequacy and appropriateness of inferences and actions based on test scores or other modes of assessment" (p. 13). Empirical evidence for consequences remains important, but additional questions are "Why?", "What is the evidence?", and "What explanations other than the intended one could explain the data?"

These are the very questions that the assessment arguments address. Warrants at each stage of the argument and in every subargument lay out the rationales for *why*. The backing marshalled at each location provides evidence for the rationales. Lines of support include theories and empirical evidence about the capabilities at issue, the situations in which they are brought to bear, actions that evince them, and criteria by which the actions can be evaluated, as well as the fit of models used to support inference.

Alternative explanations are just as important. As Cronbach (1988) put it, validation "is best guided by the phrase 'plausible rival hypotheses.' ... The advice is not merely to be on the lookout for cases your hypothesis does not fit. The advice is to find, either in the relevant community of concerned persons or in your own devilish imagination, an alternative explanation of the accumulated findings; then to devise a study where the alternatives lead to disparate predictions." *By providing tools to analyse detailed data about the processes by which students solve tasks, computational psychometrics can help assure that students are in fact acting in ways that accord with the intended argument* (Ercikan & Pellegrino, 2017; Zumbo & Hubley, 2017).

Messick (1989) described two principal threats to the main argument as "construct under-representation" and "construct-irrelevant sources of variation." That is, to what extent do tasks fail to call upon resources that are critical to the

[1] See Markus and Borsboom (2013), for a comprehensive discussion of the history and alternative views of validity in educational and psychological testing.

intended interpretations and uses? To what extent do they demand resources that are extraneous, but can hinder the examinees? Note that the more complex an assessment is, the more construct-irrelevant demands it is apt to pose.

The same validity issues apply to every stage in subarguments for interpreting features of actions and situations: What does the information coming in to that stage miss that is important, and what does it include that may be misleading? Which subgroups of examinees and what atypical patterns of performance most usefully challenge our assumptions and our methods?

2.4.2 Reliability

Haertel (2006) wrote that "The concern of reliability is to quantify the precision of test scores and other measurements" (p. 65). Two related conceptions of reliability appear in educational measurement, both readily extended to computational psychometrics: reliability as replicability (Brennan, 2001), and model-based characterizations of information and uncertainty.

Reliability as Replicability The variation among measurements of an attribute by a procedure, whether it be of persons, situations, or performances, quantifies uncertainty associated with any one measurement. The idea can be extended to multiple measurement occasions, judges' ratings of a performance, and actors portraying the same case in a standardized-patients examination. Generalizability theory (Cronbach et al. 1972) provides machinery to sort out and quantify the variance in scores associated with multiple facets of observation such as these, and their effects on inference.

Reliability-as-replication approaches are most straightforward for analyzing the evidentiary characteristics of procedures that produce values for the same variables for all examinees (or large-enough subgroups to calculate and examine by group), such as "number of strategies used" or "efficiency of solution" in a troubleshooting task. Standardized tests have this property by design. The more complex an assessment is, the more design alternatives there will be for performance environments, features of task situations, affordances students have, characterizations of performances and/or situations, and analytic approaches, and, when analyses are hierarchical, choices about analytic approaches at multiple levels. All are sources of variation that are worth examining, and all represent choices that could be optimized using metrics for evidentiary value, especially for higher-stakes uses.

Model-Based Characterizations of Uncertainty A feature of interactive assessments that hinders reliability-as-replication is that examinees can follow different paths and strategies. Some examinees may provide more data than others, and may provide evidence about different aspects of their capabilities. Model-based strategies can be employed to the weight and direction of evidence. The model posits a probability distributions for certain kinds of observable behavior conditional on

value(s) of students' latent proficiency variable(s). When performance is observed, belief is updated about their values; standard errors of measurement or posterior distributions convey the evidentiary value of the observations. The machinery of measurement can thus be employed for managing evidence and inference in complex, interactive situations.

2.4.3 Generalizability

Beyond the variants-of-procedures sense of generalizability discussed above, another sense of generalizability concerns interpretations and uses beyond assessment situations themselves. Recall first that many LCS patterns are involved in every situation, including both assessments and situations involved in extrapolation inferences. The *possibility* of valid assessment interpretations and uses is warranted by shared LCS patterns among those situations, and the fact that people may have developed resources attuned to them when encountered in those situations. But the fact that a person performs well in one situation does not assure she will perform well in another situation which from an assessor's perspective shares targeted LCS patterns. He may have developed resources that are activated in the assessment situation, effective, and consistent with the construct the assessor has in mind, but are not relevant or effective in other situations that the interpretations are meant to support.

The more complex a situation is, the more LCS patterns are involved and the more resources of more kinds a person must employ in performances. Interpretations in complex assessment situations, therefore, no matter how sophisticated the tasks and analyses, are not warranted by shared LCS patterns alone. An in-depth simulation task on troubleshooting an aircraft engine can provide vast amounts of information on how a student has solved this problem. How much does it tell us about other problems in the same engine? How about in different types of engines? How about problem-solving in different domains? Empirical evidence is required.

2.4.4 Comparability

In an application where assessments are used to compare examinees, such as licensure and admissions, are comparisons across test forms justified? The classical solution in large-scale testing programs is designing parallel test forms, followed by minor adjustments from population equating studies. In adaptive testing, the IRT model and item-selection algorithms warrant that evidence is gathered about the same capabilities and performance of different examinees is mapped into the same scale, even when they are administered items that differ in difficulty. The algorithms can be designed to obtain targeted amounts of evidence for different examinees, such

as for all to have the same amount on information or for more precise near decision points.

The situation is more complicated when assessments are interactive, require complex performances, engage multiple capabilities, or, like collaboration, involve other persons. Latent-variable modeling at higher levels of evidence accumulation provides some tools for these situations (Chap. 7). As with IRT, more complex models with latent variables for targeted aspects of proficiency provide for a common framework of characterizations of students, even when they take different tasks or follow different pathways within tasks. This is a weaker sense of comparability: The same "what" is being measured in terms of the proficiency space, but differences can exist in "how" and "how much." If one requires similar amounts of evidence for designated proficiency variables, real-time adaptation of tasks can be effected.

2.4.5 Fairness

The term fairness covers a lot of territory in assessment design, analysis, and use. I will limit my comments here to the technical sense of fairness in the *Standards for educational and psychological measurement* (AERA/APA/NCME, 2014):

> [Chapter 3 of the Standards] interprets fairness as responsiveness to individual characteristics and testing contexts so that test scores will yield valid interpretations for intended uses. A test that is fair within the meaning of the Standards reflects the same construct(s) for all test takers, and scores from it have the same meaning for all individuals in the intended population; a fair test does not advantage or disadvantage some individuals because of characteristics irrelevant to the intended construct. (p. 50)

Reducing the features of environments and affordances for response that disadvantage examinees differentially remains a sometimes-useful tactic with complex performance tasks, as to cultural differences, physical and cognitive capabilities, background knowledge, and familiarity with task interfaces and expectations.

However, as tasks become more interactive, involve more LCS patterns, and are administered to more diverse populations, the strategy of designing standard tasks becomes less satisfactory. Rather than achieving fairness by full standardization, one may employ principled variations of task schemas such that for each examinee, the main sources of challenge are construct-relevant while construct-irrelevant demands are minimized (Mislevy et al., 2013). The key is that selecting or constructing a variant in light of each examinee's profile of construct-irrelevant capabilities, so that the form of the task they encounter leverages the capabilities she brings and avoids or supports those she lacks. Alternate forms of tasks matched to students in this way can be less comparable on the surface, but more comparable in terms of the evidence they provide. Note that this is a critical use of "data of the third kind."

2.5 Conclusion

Tremendous changes are revolutionizing educational assessment. With advances in our understanding of the cognitive and social dimensions of learning, we have greater demands for assessments that are more interactive and involve more aspects of capabilities. With advances in technology, we can leverage digital environments to make these kinds of assessment affordable, and tailored to the times, the locations, and the students for whom the information they provide is most useful. With advances in statistical modelling and data analytics, we can design, gather, analyse, visualize, and act on assessment data more rapidly, and in increasingly timely feedback cycles in learning contexts.

Computational psychometrics, broadly conceived, encompasses not only the analytics of given data, but the larger contextualization of all this activity in social and historical situations. It is through methodologies such as the ones discussed in this volume that we can better understand, better employ, and better contextualize new forms of assessment. The brief sketch of sociocognitive principles helps us fit our design and our modelling efforts to the nature of learning and to the uses to which assessments are employed. Arguments for assessment design, interpretation, and use help us understand the roles of the wider range of computational methods we can now bring to bear. The social values of reliability, validity, comparability, generalizability, and fairness remain guiding stars. We must ensure them as appropriate in whatever ways they are needed, with whatever analytic methods we may devise, in whatever forms and contexts an assessment will be used.

References

American Educational Research Association, American Psychological Association, National Council on Measurement in Education (AERA/APA/NCME). (2014). *Standards for educational and psychological testing*. American Educational Research Association.

Behrens, J. T., Mislevy, R. J., DiCerbo, K. E., & Levy, R. (2012). An evidence centered design for learning and assessment in the digital world. In M. C. Mayrath, J. Clarke-Midura, & D. Robinson (Eds.), *Technology-based assessments for 21st century skills: Theoretical and practical implications from modern research* (pp. 13–54). Information Age.

Brennan, R. L. (2001). An essay on the history and future of reliability from the perspective of replications. *Journal of Educational Measurement, 38*, 295–317.

Conati, C., Gertner, A., & VanLehn, K. (2002). Using Bayesian networks to manage uncertainty in student modeling. *User Modeling and User-Adapted Interactions, 12*(4), 371–417.

Cronbach, L. J. (1988). Five perspectives on validity argument. In H. Wainer (Ed.), *Test validity* (pp. 3–17). Erlbaum.

Cronbach, L. J., Gleser, G. C., Nanda, H., & Rajaratnam, N. (1972). *The dependability of behavioral measurements: Theory of generalizability for scores and profiles*. Wiley.

Desmarais, M. C., & Baker, R. S. (2012). A review of recent advances in learner and skill modeling in intelligent learning environments. *User Modeling and User-Adapted Interaction, 22*, 9–38.

Ercikan, K. A., & Pellegrino, J. W. (Eds.). (2017). *Validation of score meaning in the next generation of assessments*. The National Council on Measurement in Educational.

Ericsson, K. A., Hoffman, R. R., Kozbelt, A., & Williams, A. M. (Eds.). (2018). *The Cambridge handbook of expertise and expert performance*. Cambridge University Press.

Greeno, J. G. (1998). The situativity of knowing, learning, and research. *American Psychologist, 53*(1), 5.

Greeno, J. G., Collins, A. M., & Resnick, L. B. (1997). Cognition and learning. In D. Berliner & R. Calfee (Eds.), *Handbook of educational psychology* (pp. 15–47). Simon & Schuster Macmillan.

Gumperz, J. (1982). *Language and social identity*. Cambridge University Press.

Haertel, E. H. (2006). Reliability. In R. L. Brennan (Ed.), *Educational measurement* (4th ed., pp. 65–110). ACE/Praeger.

Holland, J. H. (2006). Studying complex adaptive systems. *Journal of Systems Science and Complexity, 19*, 1–8.

Kane, M. T. (1992). An argument-based approach to validation. *Psychological Bulletin, 112*, 527–535.

Kelley, T. L. (1927). *Interpretation of educational measurements*. Macmillan.

Khan, S. M. (2017). Multimodal behavioral analytics in intelligent learning and assessment systems. In A. A. von Davier, M. Zhu, & P. C. Kyllonen (Eds.), *Innovative assessment of collaboration* (pp. 173–184). Springer.

Kintsch, W. (1998). *Comprehension: A paradigm for cognition*. Cambridge University Press.

Markus, K.A., & Borsboom, D. (2013). *Frontiers of test validity theory: Measurement, causation, and meaning*. Routledge.

Messick, S. (1989). Validity. In R. L. Linn (Ed.), *Educational measurement* (3rd ed., pp. 13–103). American Council on Education/Macmillan.

Messick, S. (1994). The interplay of evidence and consequences in the validation of performance assessments. *Educational Researcher, 23*(2), 13–23.

Mislevy, R. J. (2018). *Sociocognitive foundations of educational measurement*. Routledge.

Mislevy, R. J., Behrens, J. T., DiCerbo, K., & Levy, R. (2012). Design and discovery in educational assessment: Evidence centered design, psychometrics, and data mining. *Journal of Educational Data Mining, 4*, 11–48.

Mislevy, R. J., Haertel, G., Cheng, B. H., Ructtinger, L., DeBarger, A., Murray, E., Rose, D., Gravel, J., Colker, M., Rutstein, D., & Vendlinski, T. (2013). A "conditional" sense of fairness in assessment. *Educational Research and Evaluation, 19*, 121–140.

Mislevy, R. J., Steinberg, L. S., & Almond, R. A. (2003). On the structure of educational assessments. *Measurement: Interdisciplinary Research and Perspectives, 1*, 3–67.

NGSS Lead States. (2013). *The next generation science standards.* Retrieved from https://www.nextgenscience.org/next-generation-science-standards

Ryans, D. G., & Frederiksen, N. (1951). Performance tests of educational achievement. In E. F. Lindquist (Ed.), *Educational measurement* (pp. 455–494). American Council of Education.

Schum, D. A. (1994). *The evidential foundations of probabilistic reasoning*. Wiley.

Sperber, D. (1996). *Explaining culture: A naturalistic approach*. Blackwell.

von Davier, A. A. (2017). Computational psychometrics in support of collaborative educational assessments. *Journal of Educational Measurement, 1*(54), 3–11.

Wertsch, J. (1994). The primacy of mediated action in sociocultural studies. *Mind, Culture, and Activity, 1*, 202–208.

Zumbo, B. D., & Hubley, A. M. (Eds.). (2017). *Understanding and investigating response processes in validation research*. Springer.

Chapter 3
Computational Psychometrics: A Framework for Estimating Learners' Knowledge, Skills and Abilities from Learning and Assessments Systems

Alina A. von Davier, Kristen DiCerbo, and Josine Verhagen

Abstract In recent years the advances in technology provided affordances for learning and assessments opportunities. In this chapter we first describe computational psychometrics as a framework for the measurement of learners' skills, knowledge, and abilities. We discuss the changes in educational measurement that led to the need for expanding the psychometrics toolbox and describe the properties of psychometric data. We then give an example of a class of models, the Dynamic Bayesian Models that encompass many traditional psychometric models and machine-learning algorithms. We conclude by emphasizing that model complexity and power need to be balanced with the responsibility for transparency and fairness towards stakeholders.

3.1 Nomen Est Omen

In traditional psychometrics, the inferences about learners' knowledge, skills, and abilities (KSA) have been constrained due to limitations in the richness and types

The R or Python codes can be found at the GitHub repository of this book: https://github.com/jgbrainstorm/computational_psychometrics

A. A. von Davier (✉)
Duolingo and EdAstra Tech, LLC, Newton, MA, USA
e-mail: avondavier@duolingo.com

K. DiCerbo
Khan Academy, Mountain View, CA, USA
e-mail: kristen@khanacademy.org

J. Verhagen
McGraw Hill Education, New York, NY, USA
e-mail: josine.verhagen@mheducation.com

© Springer Nature Switzerland AG 2021

A. A. von Davier et al. (eds.), *Computational Psychometrics: New Methodologies for a New Generation of Digital Learning and Assessment*, Methodology of Educational Measurement and Assessment, https://doi.org/10.1007/978-3-030-74394-9_3

of data that could be collected and processed (see the Introduction and Chap. 1, this volume). There is a growing need for assessment (psychological or educational) and for training and learning tools that capture a broad range of subjects' behavior to measure complex skills such as problem solving, communication and collaboration, or learners' cognitive thought processes and non-cognitive behavior like engagement, motivation, persistence and affective (emotional) state. These tools use digital interfaces that enable rich, immersive interactions and can capture learners' data in a multimodal format that reflects real-world activities. Examples of these types of data are those from mouse clicks, keystrokes, audio and video of characters or learners interacting with each other, and the data from the interaction with simulation tasks. Inexpensive and ubiquitous sensors like webcams and microphones enable comprehensive sensing of the learners' behavior with real-time capture and recognition of the users' gaze, facial expressions, gestures, body pose, spoken responses and speech tone; this data helps provide tailored instruction to the needs of each learner. Nevertheless, reckoning from micro-features of performance to higher-order inferences about learners' capabilities can become quite challenging. Advances in fields such as machine learning, educational data mining, computer vision and natural language processing are making it possible to analyze and model a much wider range of evidence for estimating learners' KSAs, and hence, the forms of assessment we can use are also expanded.

Building on traditional psychometrics, computational psychometrics has incorporated techniques from the fields above to analyze large-scale/high-dimensional learning, assessment, biometric, or psychological data and provide actionable and meaningful feedback to individuals based on measurement of individual differences as they pertain to specific skills and areas of enquiry. These competencies may include a wide variety of cognitive and non-cognitive skills; twenty-first century competencies like communication, collaboration, problem solving (e.g. diagnosing patients), troubleshooting (e.g. computer networks etc.) and related workforce readiness skills.[1]

The term *computational psychometrics* refers to the blend of computational techniques and traditional psychometric theory in order to answer the psychometric questions arising from the changing landscape of education or psychology, the expansion of the definition of psychometric data, and the changing of assessment needs. See the work of Cipresso and his colleagues (Cipresso et al., 2017) for psychological applications.

[1] Or the skills for a new economy (SNE): https://www.networksnorthwest.org/talent/job-seekers/career-spotlight/career-skills-for-the-new-economy.html

3.1.1 What Is Computational Psychometrics?

A New Terminology Almost everyone agrees that there is an urgent need to expand the methodologies in existing psychometrics to accommodate the challenges from the new forms of learning and assessment tasks in digital world. To help researchers in the field to easily communicate ideas along this line, it will be tremendously helpful if we can assign a concise name to this new branch of psychometrics. Coming up a concise term to encompass a discipline that has such a vast intension is extremely challenging. After years of back and forth, we noted that despite the wild differences among the new methodologies, they share one common feature, computational models. As such, we arrived at the term Computational Psychometrics (von Davier, 2015), by which we want to capture the most essential feature of this new subdiscipline, i.e., data-driven computational algorithms, while keep its primary alignment to psychometrics. This may not be a perfect name, as always in the history of science, but it is a concise one that highlights the key features of this new subdiscipline. One of the goals of this book is to further define the scope of this new subdiscipline through more specific examples, by which we hope to help researchers working in the forefronts of psychometrics to communicate ideas more clearly and concisely.

Computational psychometrics joins other new disciplines, such as computational social science, computational statistics, etc. In all of these newly minted disciplines, the emphasis is not on computer-facilitated analyses (which has been happening for the past 60 years or so), but on data-driven algorithms that create, augment, or even replace the theory-driven disciplines.

A New Framework Specifically, computational psychometrics is defined by its reliance on psychometric theory for characterizing higher-level constructs, which are designed to support the understanding and generalization of learners' performance across multiple rich activities in an interactive (digital) environment. In technical terms, computational psychometrics is a hierarchical inference model, with stages in evidence-identification, in a multi-staged reckoning from low-level data (pixels, keystrokes) to high-level constructs that comprise the psychometric models (see von Davier, 2017). These features lead to an approach to data collection design that is oriented towards eliciting the appropriate evidence from these fine-grained data, extending, therefore, the Evidence Centered Design approach, and towards the consideration of techniques from data mining and machine learning fields that identify patterns in the fine-grained data, which can be included as part of the multi-staged inference identification.

To provide an analog to traditional psychometrics, we mention generalizability theory (Cronbach et al., 1972): In a classical test theory framework, generalizability theory incorporates the effects of different sources of variance (task specific, environment specific, timing, scoring, etc.) into the measurement model. The computational psychometrics framework incorporates these ideas of generalizability of inferences across complex multimodal data.

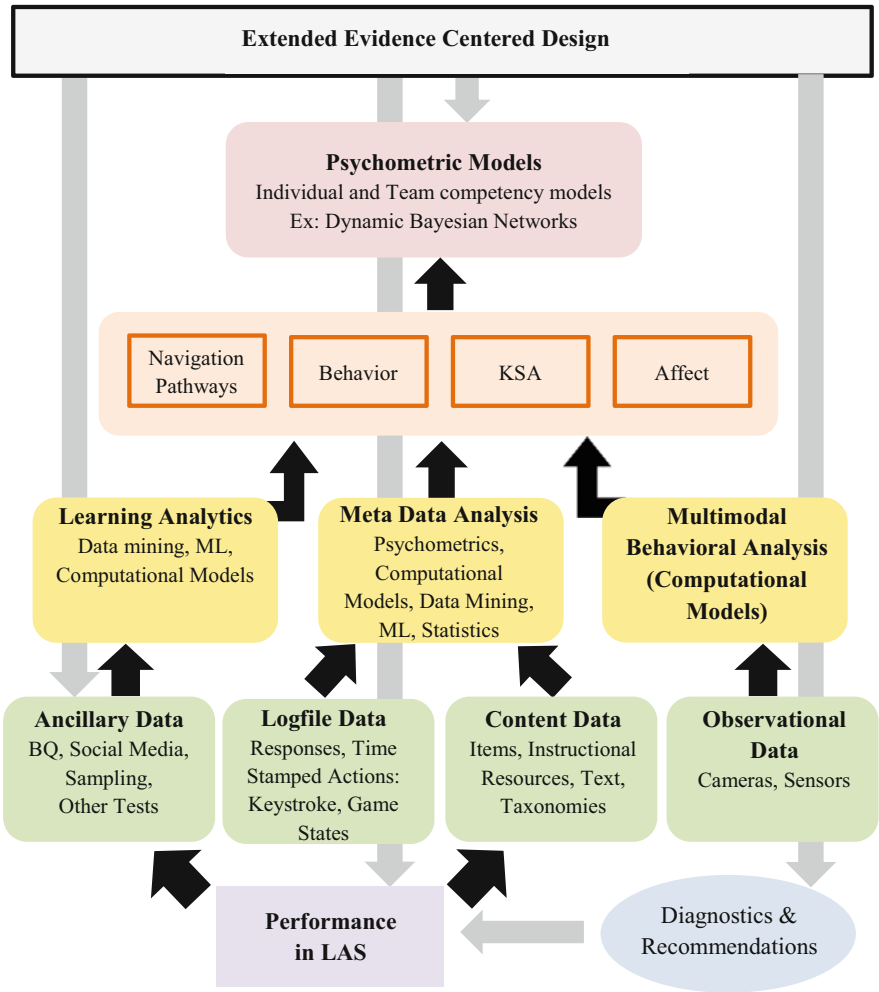


Fig. 3.1 Computational psychometrics framework for the measurement of KSAs, Behaviors, Affect, and educational Pathways in a complex virtual learning and assessment system (LAS)

Computational psychometrics originated with the analysis of process data (the multimodal data from the process of interacting with a performance task, a collaborative task, or with a learning system). Computational psychometrics really started to flourish once the power of cloud computing, together with the alignment of databases, and improved data governance became available for real time applications of computational models to the analysis of these data.

In Fig. 3.1 we illustrate that computational psychometrics is intrinsically linked to data: (a) multiple data sources, (b) data with dependencies among variables, (c) data that is primarily collected according to an (extended) evidence-centered

design, but also data that is a by-product of the complex interactions (say, keystroke data), (d) data that needs to be structured into databases that are linked and aligned according to the variables of interest (common identity management, construct of interest, etc.), and (e) multimodal data which is jointly analyzed for the measurement of learners' KSA. Specifically, in Fig. 3.1, we show how the top-down approach (the data collection is designed following a theory, the data is well-structured, and a relational psychometric model is hypothesized) and the bottom-up approach (the rich data that is available as a by-product of the interaction with a complex virtual system, as well as other data that may be available, such as demographics, school data, census data, etc.) blend together and allow for the data to be analyzed jointly to infer KSAs. While the availability of aligned data sets from multiple sources is not yet a reality for many institutions and applications, it is becoming available for more and more groups. In these pockets of reality, learning analytics based on ancillary data from schools and content data (as text or video from the instructional content) have been merged with the learner's process and outcome data, often run all in one analysis—if the data at the school level, at the testing institution level, and at the learner level is aligned (as in the data cube example—see below for details).

In the rest of the chapter, we describe the changes in educational measurement that led to the need for expanding the psychometrics toolbox, followed by a description of the properties of psychometric data, and then we give an example of a class of models that encompass many traditional psychometric models and machine-learning algorithms, and which are often considered for modelling data from game-based assessments. This is the Dynamic Bayesian Modeling (DBM) modelling approach. We conclude with a reminder that model complexity and power need to be balanced with the responsibility for transparency and fairness towards stakeholders.

3.2 Changing Nature of Educational Measurement

Traditionally, educational measurement meant assessment and assessment has been synonymous with "test." Nowadays, educational measurement includes assessments, but also holistic learning (learning of academic and non-academic skills), and navigation support (providing learners with recommendations for their education pathways), as it is being addressed currently in the education community. See the work on frameworks such as of Camara et al. (2015) that include social-emotional learning and assessment, research on learning and assessment systems (von Davier et al., 2019a, b), and the work on game-based assessment of DiCerbo and Behrens (2012).

In this section, we will discuss the changes in educational measurement, with an emphasis on assessment, because assessment has been the core of educational measurement. Tests were time-bound activities in which learners typically answered questions. Questions were written specifically to assess one and only one construct.

The answers were then scored as correct or incorrect. Psychometric models, including classical test theory (CTT) and item response theory (IRT) were created largely to address these dichotomously scored responses.

As with everything else that depends on technology, assessment opportunities have changed quickly over the past few years. While periodical summative and formative assessments have traditionally been the main sources of data to evaluate a learner's KSA or a learner's learning progress, digital environments provide the opportunity to continuously monitor and support learner's performance for the duration of the interaction with the system at a level of detail and a scale that were not possible previously. Traditional psychometric models are not well suited to model continuous growth, as the most basic assumptions underlying these models (unidimensionality of the latent construct, stationarity of the ability during a full assessment, local independence) are violated. Over time, numerous extensions of the traditional models have been proposed in order to deal with one or more of these violations and, as we indicate later, many of these extensions can be used with the rich data from digital education.

Digitally-based testing allows for the development of rich and technology enhanced activity types that go beyond the limited format of multiple-choice or open-ended written responses; these new types of tasks can potentially be more reflective of real-world situations (e.g. solving problems in a digital environment) and allow for the measurement of inquiry skills or synthesis ability, hence providing a richer measurement of the learners' abilities. To evaluate those tasks, multiple sources of evidence can be collected that go beyond the right/wrong evaluation of an answer; for example, one can presumably better identify the strengths and weaknesses of learners by taking into account the timing and sequence information from learners' interactions with the tasks and subtasks, or information surrounding specific mistakes a learner has been making along the way. Although there is a history of such tasks in assessment (see, for example, Ryans & Frederiksen, 1951), technological limitations confined their use to a small number of costly hard-to-scale applications, and local idiosyncratic uses as in classroom and projects and role plays.

The larger range of activity types possible due to emerging technology offers opportunities to develop digital assessments that can support a broader range of skills, such as collaboration, critical thinking, and creativity. While it is difficult to elicit evidence of these constructs with a traditional multiple-choice test, we can elicit a much wider variety of behavior, and thus evidence, in a digital learning environment. Digital forms of interaction, data capture, and analysis remove previous barriers to a greater variety of forms, contexts, purposes of assessment.

We can describe the evolution from technology-enhanced assessment to assessment-enhanced technology as passing through four stages (DiCerbo & Behrens, 2012). The first is the use of technology to create new item types and improve feedback on assessments that are still close neighbors of pre-digital "paper-and-pencil" tests. The second level is defined by the creation of new assessments that are focused on a central, context-providing problem to be solved. These might be simulations that are designed and administered as discrete assessment activities.

The third level is the creation of natural digital experiences designed with embedded assessment arguments. This would include the use of stealth assessment in a game where learners are playing the game, but information about constructs of interest is collected as learners are engaged in the activity (Shute & Ventura, 2013). Finally, the fourth level consists of an ecosystem in which information from a variety of natural digital experiences like those in level three is accumulated. In this case, the activities may or may not have been designed as assessment activities.

As the field of psychometrics moves to support each level, the entire process of assessment development must be reconsidered. Evidence-Centered Design (ECD; Mislevy et al., 2003; Arieli-Attali et al., 2019; Chap. 1 in this volume) provides a framework by which to consider each element of the process. Specifically, it provides the concepts of the learner model, task model, and evidence model. The learner model defines the constructs about which we want to make inferences, for example, it might include the skills and subskills for critical thinking. Next, the task model defines the activity types that will elicit evidence. The activity could be questions, a simulation, a game, or any other activity a learner might engage in. The evidence model specifically calls out evidence identification (the extraction of key elements from the work product produced by the learner) and evidence aggregation (the accumulation of information into a score via a statistical model).

Traditionally, this system has been developed based on theory. The learner model is often based on a research-tested theory about how skills develop and relate to each other. The tasks and activities are specifically developed to elicit evidence that test developers theorize tells us about the constructs of interest. In game-based assessments developed using ECD, the game play actions thought to be evidence of the construct are hypothesized and tested during development of the game. For example, in the creation of SimCityEDU (see Mislevy et al., 2014), designers started with a learning progression for the construct of systems thinking that described levels of increasing sophisticated understanding, from lack of understanding of how variables in a system influence each other, to univariate understandings, to multivariate understandings in which actions can have multiple consequences and results can have multiple causes. They then designed experiences that would provide evidence about the level at which a learner functioned. For example, they hypothesized that if a player observed high air pollution and bulldozed a coal plant without placing another energy source, the player had a univariate understanding that the coal plant produced air pollution but not the multivariate understanding that it produces air pollution and powers the city. The game scenarios were created specifically to produce evidence to make inferences about systems thinking.

However, some see a digital world in which there is so much data occurring from people's interactions in "natural" digital environments that there is not a need for this careful a priori specification of evidence. Instead, the idea is that machine learning, and eventually artificial intelligence, can be used to identify the key pieces of evidence, and their combinations, to produce scores. Currently, this is a reality only for a minority of projects, for a variety of reasons. Many of these techniques require some sort of ground truth or "gold standard" to train the model. If we are talking about digital assessments, then it is not obvious what this standard is. Is

it a traditional assessment? That violates the argument that these new forms of assessment will improve on traditional assessment. Is it an educator's judgment? Part of the lure of educational technology is that it can scale beyond what an individual educator can observe. Nevertheless, see Hao et al. (2017), Halpin et al. (2017), Shu et al. (2017), and Stoeffler et al. (2018) for examples of studies where the authors proposed methodologies that can contribute to scoring. While the applications vary, the methods proposed in these studies use a blend of theory and data. Second, there are often large jumps required between observed action and inference about knowledge and skills. In the SimCityEDU example above, the connection between bulldozing power plants and systems-thinking was further investigated through play testing and cognitive labs, in which learners thought out-loud as they played. It is not clear that considering evidence pieces simply because they appear to correlate with other evidence pieces is sufficient evidence for their inclusion in a model. The systems and projects for which the knowledge discovery from the data approach seems to work are the learning systems, where the ground truth can be provided by educators' input, traditional assessments and GPAs.

Currently, it appears that a combination of both human-driven and data-driven approaches may be most successful in moving towards the goal of improved assessment, hence computational psychometrics. For example, in the SimCityEDU work, exploratory data analysis techniques were used to identify unexpected or unhypothesized evidence (DiCerbo et al., 2015). The Bayesian network models used to aggregate information were also a combination of human-created priors updated as data from game players arrived (Castellano et al., 2015). In a more traditional data mining approach, Sao Pedro et al. (2012) compared the performance of models developed from machine learning alone versus those with human-enhanced machine learning. Rather than inputting all potential evidence pieces into the machine learning algorithm, the researchers based the included features on their relevance to the construct, as rated by a domain expert. This human-influenced model performed better than the machine-learning-only model in predicting learners' scores. As an added benefit, intelligibly construct-relevant features of performance can ground local feedback to students to support further improvements.

Recent advancements in the fields of data science, machine learning and artificial intelligence and the significant boost in computational-power and in the exponential growth of the size of the data, together allow for novel ways of making sense of the stream of data coming from digital environments. These analyses and results can support inferences about constructs of interest or learning processes and augment the theory-driven methods. However, we want to consider the entire assessment and learning processes as we explore these developments. The most sophisticated data analysis techniques will not be able to create accurate solutions if there is no signal to be found—big data is not necessarily good data. Also note that the current reality of assessments is that the amount of data available is nowhere near the amount needed to take full advantage of many of the machine learning techniques, and the amount of evidence in that data about any particular construct is often even smaller. Again, for learning systems or for sensor-based evaluations (such as video-interviews or multimodal assessment of teamwork) the data is rich, but there is still

work to be done to best identify the relevant features. As mentioned above, it appears that a combination of human theory and these sophisticated methods may yield the best results.

In analyzing these virtual tasks, the focus is now on how the information in a given task performance generalizes to inferences about learners' skills potentially displayed in tasks that differ in various ways. This perspective is relevant for assessments involving higher-level skills and complex performance tasks, which are set in a specific context and draw on particular knowledge and representations, such as game-based assessments or simulation-based assessments. The transfer of the behavior of the learner in one game may or may not occur in a different game or in a classroom situation (Gao et al., 1994). In order to ensure that the data from one authentic performance task is generalizable to other performances, one needs to consider the design of the tasks so that the claims to be supported go beyond the specific task and its context (Arieli-Attali et al., 2019) and to empirically replicate the results. In other words, we are less interested in the reliability of a measure of a skill in a particular context, but more in the generalizability of the results for the inferences we need to make. In a classical test theory framework, generalizability theory has provided a framework for incorporating the effects of different sources of variance (task specific, environment specific, timing, scoring, etc.). The computational psychometrics framework incorporates these ideas of generalizability of the inferences across complex multimodal data, often in a Bayesian context.

In sum, recent developments in educational practice, changes in assessment needs, availability of large and curated data sets, and unprecedented computational power demand a new set of psychometric tools. Computational psychometrics blends computational (e.g. machine learning and multimodal analytics) methods and expert human intelligence, to address the psychometric questions raised by these new developments.

3.3 Psychometric Data

Computational psychometrics is intrinsically linked to data, its availability, governance, size and quality. This is why we included a section on psychometric data in this chapter.

Despite the advances in educational technology and in the methodology and even in the availability of large data sets collected at each administration, the way assessment data is collected, stored, and analyzed by testing organizations is not conducive to these real-time, data intensive computational methods that could reveal new patterns and information about learners.

In their recent paper, von Davier et al. (2019a) propose a new way to label, collect, and store data from large scale educational learning and assessment systems (LAS) using the concept of the "data cube." A data cube is a multi-dimensional data structure model for storing data in the data warehouse. A data cube represents

data in terms of dimensions and variables. Dimensions in a data cube represent characteristics in the data set (people, objects of interest, variables, etc.). While the approach proposed by von Davier et al. (2019a, b) for educational data is inspired from database science and data governance, and it can be seen as an extension of Cattell's three-dimensional data box for longitudinal analyses and time series in which each direction represents a different dimension of the data: persons, variables, and time (Cattell, 1966). Cattell's data box has influenced the categorization and understanding of various psychometric analyses. For example, the variable dimension focuses on group-level relationships between variables (intersubject variability), utilizing analysis of variance (ANOVA), multiple regression, and structural equation modelling, and other methods that use persons x variables data. The time dimension allows to focus on the pattern of change in an individual over time (intrasubject variability). Methods include autoregressive integrated moving average (ARIMA) models, dynamic factor analysis, and state space modelling. One of the characteristics of the data cube presented in von Davier et al. (2019a, b) is that the data cube for educational data should also allow for storing the content for tests (items, or more complex activities such as investigations that might be better described as "tasks") and instruction (videos, simulations, items with scaffolds) as another dimension in the cube, which opens-up new avenues for personalized learning by "dicing" the cube on person x content x time. While nowadays other techniques are used for data governance, techniques that can scale up better and faster, the data cube provides a clear concept for understanding the data structure that supports many types of analyses. This enterprise data paradigm makes the application of machine-learning, learning analytics, and complex analyses possible.

von Davier et al. (2019a, b) also argue for the adoption of data exchange standards to support the alignment of different databases. For example, the IMS Global Learning Consortium[2]has a set of standards that are relevant to learning and assessments with technology-enhanced items. One of these standards, the Caliper standard, addresses event data from the process of learning or of interacting with a performance-based task.

There are several points we want to make about the data collection and design: (a) The complexity of the data and the demands of the applications lead to more complex models that use computational features to reduce the dimensionality of the data and extract new patterns from the data. (b) There is data that has not been traditionally associated with psychometrics. Nevertheless, the power of ML-based data discovery lies in the alignment of databases; hence the computational models thrive within a data cube, where many of the possible dimensions of the data have been considered and made available for the analyses (using data exchange standards, for example). (c) Computational psychometrics is not a set of tools for the analysis of process data, but rather a paradigm for the inclusion and analysis of all relevant data associated with the (psychological/social/educational) phenomena of interest (see also Fig. 3.1).

[2]http://www.imsglobal.org/

3.4 An Example of Measurement Models Class for All Types of Data: Dynamic Bayesian Models

In this chapter we will only give an overview of one class of models that illustrates well how psychometric models and computational models can blend. We chose the Dynamic Bayesian models (DBM) because of their flexibility to address many of the measurement needs and data features encountered in complex learning and assessment systems. DBMs are a family of general dynamic models that can reduce to simple IRT models in some instances (one stationary construct, no dependences), or to Bayesian networks (BN) for stationary constructs with dependences, or they become equivalent to many other models such as (hidden) Markov models and Bayesian knowledge tracing (BKT) in complex measurement settings, such as learning or collaboration. The DBM are dynamic models suitable to model changes in the variables of interest, such as learning. Moreover, the DBM are also an essential part of the ML tools, and the Bayesian framework is instrumental to all high-efficiency ML algorithms, such as deep learning (see Huang et al., this volume). The other chapters of the book present other computational models.

There are two main challenges when applying psychometrics to continuous data from complex data sources. The first challenge is how to use data from multiple sources to extract and quantify evidence of ability or skill mastery to use for assessment. The second challenge is to model the dependencies in the data over time. These challenges result in violations of important assumptions (unidimensionality of the latent construct, stationarity of the ability during a full assessment, local independence) underlying traditional psychometric models.

There is a growing literature on using DBMs for more complex forms of assessment using ECD, addressing the first challenge. The dynamic component is particularly suited to address the challenge of modeling situations with ongoing learning, where the KSAs are expected to change (see Chaps. 1, 3, and 4, this volume). DBM applications result in a probabilistic quantification of underlying states, which can be the probability of mastering a skill or a probability distribution expressing the ability level of a learner. DBMs model the transition between those states over time.

Bayesian inference allows for the modular assembly of analytic models, as it provides a mechanism to combine model components linked by conditional probability (Chap. 6, this volume). This mechanism can be used to combine evidence from various and complex sources with conditionally defined relationships, or to combine evidence from consecutive time points. An additional advantage of Bayesian inference is the option to include prior information, which enables information from outside the current learning environment (e.g. demographics, or information from another learning environment or test) to inform KSAs. This can substantially reduce the time necessary to arrive at accurate KSA estimates, as well as enhance inference when only little information about a learner is collected at the current measurement moment. Finally, the posterior distribution of a KSA provides an intuitive metric of uncertainty around the KSA estimate of interest.

An overview of dynamic DBMs in educational measurement to date is given in Reichenberg (2018) and Levy and Mislevy (2016). See also Deonovic et al. (2018), where models for learning and assessment are nicely explained and connected. Suppose we collect data from a learner in a learning environment or educational game from two consecutive measurement moments: $t = 1$ and $t = 2$. The observations X at these two measurement moments can range from one correct or incorrect answer on a question, to evidence based on a vector, matrix, or cube of data from the learner interacting with the learning environment.

Let θ be a (vector of) knowledge, skill or ability we would like to measure based on X for each of the learning sessions; we want to calculate $p(\theta_{t=T}|X)$, the learner specific posterior distribution at the current time point.

$$p(\theta_t|X_t) = \frac{p(\theta_t, X_t)}{p(X_t)} \propto p(\theta_t)\, p(X_t|\theta_t)$$

Here, $p(X_t|\theta_t)$ represents the measurement model component that provides the conditional probability distribution of the observations X at time point t as a function of all possible values of KSAs, also known as the likelihood function. And $p(\theta_t)$ is a prior distribution where $p(\theta_{t=1})$ may represent our beliefs about the learner's KSA distribution before we have observed any evidence in the current learning environment.

The joint distribution of observations and KSA's at each of both measurement moments can be written as a product of conditional probability distributions representing different model components:

$$p(\theta_t, X_t) = p(\theta_t)\, p(X_t|\theta_t)$$

For the two measurement times $t = 1$ and $t = 2$ where we assume that the evidence at time 2, $X_{t=2}$ depends only on the ability $\theta_{t=2}$, this joint distribution simplifies as

$$p(\theta_{t=1}, \theta_{t=2}, X_{t=1}, X_{t=2}) = p(\theta_{t=1})\, p(\theta_{t=2}|\theta_{t=1})\, p(X_{t=1}|\theta_{t=1})\, p(X_{t=2}|\theta_{t=2})$$

The measurement model $p(X_t|\theta_t)$ can be, for example, a traditional psychometric model like IRT, which defines the likelihood of a set of binary answers on test items as a function of values of a latent continuous ability. Extending the model to more than two measurement moments can be achieved by chaining models for consecutive measurement moments together, assuming that information from all previous measurement moments is contained in the last previous measurement moment.

The measurement model can also consist of evidence rules (generated by machine learning techniques or defined by experts) that reduce a complex pattern of observations into a (set of) value(s) of the likelihood of which can subsequently be defined as a function of (a vector of) KSA values. For example, such a function

could be the probabilities of belonging to a each of a set of classes or clusters as predicted by a machine learning algorithm.

Here $p(\theta_{t=1})$ and $p(\theta_{t=2}|\,\theta_{t=1})$ can be considered prior probability distributions for the KSA at $t = 1$ and $t = 2$, respectively. For $p(\theta_{t=1})$, this distribution can represent our beliefs about the learner's KSA distribution before we have observed any evidence in the current learning environment, as mentioned before. A commonly used default population distribution represents what is known before any responses are collected from a learner; for example, the likelihood of their KSA/proficiency corresponds to the frequency of that KSA in the population. However, there are many options to include additional information in the prior distribution, such as the distribution of the KSA conditional on age. In case of multiple KSAs, the model can represent complex hierarchical or prerequisite relationships between the KSAs, and the estimates of different but related KSAs can inform the prior distribution. If accurate, a more informative prior will make the KSA converge faster to the true value. If it is not accurate, then the model may result in biased estimates. In case of multiple KSAs, the model can represent a complex hierarchy of prerequisite relationships between the KSAs, and the prior information of different but related KSAs can inform the estimation.

Here $p(\theta_{t=2}|\,\theta_{t=1})$ represents the prior for the KSA at moment $t = 2$, conditional on the KSA at $t = 1$, or the "transition model". This is where the dynamic aspect of dynamic Bayesian networks comes into play and where the opportunities to move from an assessment to a learning model arise. In general, there are two approaches to specifying the transition model. The first approach (also called "fading") focuses on adding uncertainty to estimates from previous measurement moments as more time passes since the previous measurement moment and/or there is additional evidence that indicates that the KSA has changed. The second approach actively models the learning process by quantifying the probability that learning or forgetting takes place over time, dependent on, e.g., practice opportunities (as BKT does), time passed, developmental expectations (e.g. growth curve models) or additional evidence that might indicate that learning, or forgetting, has taken place. Even though many widely used algorithms for adaptive learning assume an active transition model, there is not much research focused on validating this part of the model and the underlying assumptions about how learning occurs over time in a given environment.

Extending the model to more than two measurement moments can be achieved by chaining models for the transition between consecutive measurement moments together, assuming that information of all previous measurement moments is contained in the last previous measurement moment (which is the Markov property of order one). However, it is possible to specify transition models that take information from the next previous measurement moments into account (which is then the Markov property of order two; see Begleiter et al., 2004).

To provide reporting or adaptivity based on a learner's KSA we often want to calculate $p(\theta_{t=T}|\,X)$, the learner specific posterior distribution at the current time point. The advantage of Bayesian modeling is that the uncertainty about the estimate is always built into the posterior distribution. Taking into account uncertainty and

changes in uncertainty in the transition model is a challenging task, but if done well it will reflect the uncertainty in the estimate of a learner's KSA/proficiency as it is changing over time.

There are two crucial elements when working with dynamic Bayesian networks. First is the definition of the time of the measurement. States can be assumed to be subject to change after each examinee response, or after a set of grouped responses (by e.g. a game round, a topic, a set time limit of a day). In general, the more examinee responses a set comprises, the more reliable the assessment of the examinee's ability at that measurement moment will be, but the less sensitive the estimate could be to changes in examinee's ability.

The second element is the definition of the transition parameter – transition model (transition matrix, for discrete KSAs), which determines how much weight is given to the previous state and which represents how likely it is that the state changes between one measurement moment and the next. This transition parameter can be set to a fixed value, modeled with an additional function (based on e.g. how much time has passed between two measurement moments with a time decay function), or estimated from the data by adding one or more additional parameters to an already fairly large set of parameters. The transition model can incorporate/reflect assumptions about growth, or only work on the weight given to previous observations.

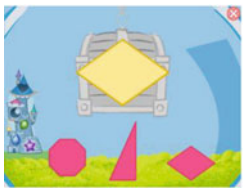
A Conceptual Example

An application which illustrates a basic dynamic Bayesian model and which uses all the elements described above is an educational game-based learning app developed by Kidaptive[3] to advance shape understanding and to teach pattern recognition for preschoolers. The two shape games (Fig. 3.2) had the learners work on shape identification and manipulation (translation, rotation, scaling, and composition), with difficulty ranging from matching similar shapes to composing an outlined shape "puzzle" from multiple pieces, dragging them to the target area and rotating them to fit. In the two pattern games (Fig. 3.3), the learners were shown a sequence of objects (such as ABAB, ABCABC, or ABBABB) and had to choose the correct object(s) to continue the pattern, with difficulty varying by pattern and the availability of distractors.

In order to apply the dynamic Bayesian model described theoretically above we make the following considerations.

Prior distributions: As preschoolers develop rapidly between 2 and 5 years old, age is a very good predictor of performance in these games. We modeled performance as specific normal distributions with half year increments based on a pilot sample: $\theta_{t=1} \sim N(\mu_{age}, \sigma_{age})$.

[3] https://www.kidaptive.com/



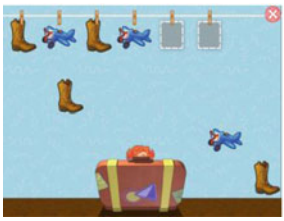
Shapes Game A Level 2

Task: Identify Advanced Shape

Shapes Game B Level 8

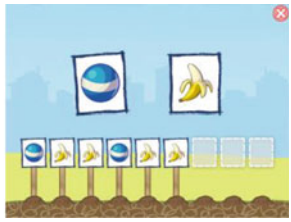
Task: Compose Shape: 3 parts

Fig. 3.2 Screenshots of the two shapes games



Patterns Game C Level 4

Task: Extend AB(_)(_)



Patterns Game D Level 7

Task: Extend ABB(_)(_)(_)

Fig. 3.3 Screenshots of the two patterns games

Measurement model: We chose a standard 1PL IRT model to model the likelihood of a correct/incorrect response to a question given the learner's ability and the item's difficulty.

Transition model: We started out with a very simple fading model, treating each "game" of 4 items as a measurement moment. At the beginning of each game, we would take the posterior distribution from the previous game as the prior distribution for the next game, however we would reset the standard deviation to a fixed and larger value. A more advanced fading model included the time information; we did some research on a time-based fading model where both mean and standard deviation would shrink to an age-based "reference distribution" as more time passes between measurement moments. An active transition model here could model an expected growth curve over time and base the prior distribution for the next measurement moment on expected growth between measurement moments.

Combining evidence from multiple sources: Since we have two games for each dimension (shape recognition/patterns), there is an opportunity to model combining evidence from multiple sources into one underlying latent skill estimate. A Bayesian dynamic two-factor model with "game scores" as indicators proved challenging to estimate but interpretable.

Since Bayesian dynamic networks grow fairly complex relatively fast, the computational algorithms necessary for estimating all interrelations until recently was an obstacle. There are promising developments however in probabilistic modeling, using some of the infrastructure behind neural networks as we can see in the next chapters.

3.5 Conclusions

While traditional psychometrics has offered a framework for designing traditional assessments (linear tests, adaptive tests, matrix-design surveys), computational psychometrics provides a framework for the next generation assessment and learning systems that include technology-based tasks that are context-rich and which provide new insights about learners' KSAs.

Technological progress has made virtual performance-based assessments, such as games and simulations, possible. Nowadays, the typical data from these complex tasks are process data that record a test taker's timestamped activities and the situational/environmental variables. Similarly, learning and assessment systems and intelligent tutoring systems record learners' interactions with the instructional material over time. While some of this additional data is collected according to the theory of learning, there is still incidental data available that should be considered. This minutia of information needs to be parsed and understood. Collaborative learning and assessment brings its own challenges, where the observations collected depend on the interactions among the collaborators in addition to the time dependencies. The chat or video data that represent the evidence for the collaboration need also to be parsed, features extracted, then these intermediate features need to be modeled and integrated in the psychometric models.

As assessment tasks become more and more technology-dependent and generate more and more process data that may provide new insights in the learners' understanding, it is necessary to extend the existing psychometric methodologies to account for this increased complexity. The extension may include both new methods for creating quantitative representations of the complex responses (such as video, audio, and keystroke), and new modeling methods to relate the complex evidentiary representations to the targeted constructs. As such, a new name, computational psychometrics (von Davier, 2015), was introduced as an umbrella term to encompass these new methodologies. Computational psychometrics merges the data-driven approaches with the theoretical (cognitive) models to provide a rigorous framework for the measurement of skills in the presence of process data.

Considerations for the next generation of psychometricians. The skills needed to parse, aggregate, and model these types of data are generally not well covered in most educational measurement programs. Moreover, for these methodologies to be successful, one also needs insight into the theory of learning, content of interest, and psychometric modeling goals. As such, psychometric researchers need to develop new skills to be able to keep pace with the increased demands for handling complex

assessment tasks in the future. It is also necessary to work in interdisciplinary teams to obtain the best results. It has been noted that in the EdTech community several learning systems have been developed outside the learning sciences and psychometric community and little attention has been given to the measurement concepts of reliability and validity. In consequence, these systems were less successful in proving their efficacy in the classroom. Similarly, learning analytics techniques applied to large school data (ancillary data in Fig. 3.1) have often lacked the theoretical foundation of learning sciences and the psychometric properties. Computational psychometrics provides a framework where these techniques from machine learning and data science can be integrated within the psychometric theory for coherence and generalizability.

Considerations on transparency. We should not lose sight of the need for educators, learners, and parents to understand the results of a learning and assessment system. Even the concept of a probability as a result is often highly confusing for many audiences, and likely to be misinterpreted (Bradshaw & Levy, 2019). One solution to difficulties with understanding these results is the creation of levels with descriptors of the modeling results. Educators and learners also want to know why they received the score they received. They want to be able to see the link between what they did and their performance summary. The more we create, or allow our statistical machinery to create, complex, opaque scoring models, the more difficulty there will be convincing educators and learners they should trust them. New computational models should be evaluated not just for their ability to accurately summarize data, but also for their interpretability to a lay audience who will be asked to make decisions based on their use.

Considerations on fairness. Last but not least, the issue of fairness and differential functioning of tasks—both for learning and for assessment— and of the training data used for the ML algorithms to extract meaningful features needs to be considered with the same care and attention it has been accorded in the past, in traditional psychometrics. Fortunately, in the past few years, the awareness around these issues led to more specifics activities and checks on the computational models (e.g., Madnani et al., 2017).

References

Arieli-Attali, M., Ward, S., Thomas, J., Deonovic, B., & von Davier, A. A. (2019). The expanded evidence-centered design (e-ecd) for learning and assessment systems: A framework for incorporating learning goals and processes within assessment design. *Frontiers in Psychology, 10*, 853. https://doi.org/10.3389/fpsyg.2019.00853

Begleiter, R., El-Yaniv, R., & Yona, G. (2004). On prediction using variable order Markov models. *Journal of Artificial Intelligence Research, 22*, 385–421. https://doi.org/10.1613/jair.1491

Bradshaw, L., & Levy, R. (2019). Interpreting probabilistic classifications from diagnostic psychometric models. *Educational Measurement: Issues and Practice, 38*, 79–88.

Camara, W., O'Connor, R., Mattern, K., & Hanson, M. A. (2015). *Beyond academics: A holistic framework for enhancing education and workplace success* (ACT Research Report Series 4). ACT.

Castellano, K., Hoffman, E., Bauer, M., Bertling, M., Kitchen, C., Jackson, T., Oranje, A., DiCerbo, K., & Corrigan, S. (2015). *Game-based formative assessment for argumentation: Mars Generation One: Argubot Academy*. Paper presented at the annual meeting of the American Educational Research Association, Chicago, IL.

Cattell, R. B. (1966). The data box: Its ordering of total resources in terms of possible relational systems. In R. B. Cattell (Ed.), *Handbook of multivariate experimental psychology* (pp. 67–128). Rand-McNally.

Cipresso, P., Bessi, A., Colombo, D., Pedroli, E., & Riva, G. (2017). Computational psychometrics for modeling system dynamics during stressful disasters. *Frontiers in Psychology, 8*, Article 1401. https://doi.org/10.3389/fpsyg.2017.01401

Cronbach, L. J., Gleser, G. C., Nanda, H., & Rajaratnam, N. (1972). *The dependability of behavioral measurements*. Wiley.

Deonovic, B., Yudelson, M., Bolsinova, M., Attali, M., & Maris, G. (2018). Learning meets assessment. *Behaviormetrika, 45*, 457–474.

DiCerbo, K. E., & Behrens, J. T. (2012). Implications of the digital ocean on current and future assessment. In R. Lissitz & H. Jiao (Eds.), *Computers and their impact on state assessment: Recent history and predictions for the future* (pp. 273–306). Information Age Publishing.

DiCerbo, K. E., Bertling, M., Stephenson, S., Jie, Y., Mislevy, R. J., Bauer, M., & Jackson, T. (2015). The role of Exploratory Data Analysis in the development of game-based assessments. In C. S. Loh, Y. Sheng, & D. Ifenthaler (Eds.), *Serious games analytics: Methodologies for performance measurement, assessment, and improvement* (pp. 319–342). Springer.

Gao, X., Shavelson, R. J., & Baxter, G. P. (1994). Generalizability of large-scale performance assessments in science: Promises and problems. *Applied Measurement in Education, 7*(4), 323–342.

Hao, J., Chen, L., Flor, M., Liu, L., & von Davier, A. A. (2017). *CPS-Rater: Automated sequential annotation for conversations in collaborative problem-solving activities* (Research Report No. RR-17-58). Educational Testing Service. https://doi.org/10.1002/ets2.12184

Halpin, P. F., von Davier, A. A., Hao, J., & Liu, L. (2017). Measuring student engagement during collaboration. In A. A. von Davier (Ed.), *Measurement issues in collaborative learning and assessment* (Special Issue). *Journal of Educational Measurement, 54*, 70–84.

Levy, R., & Mislevy, R. J. (2016). *Bayesian psychometric modeling*. Chapman & Hall/CRC. https://doi.org/10.1201/9781315374604

Madnani, N., Loukina, A., von Davier, A. A., Burstein, J., & Cahill, A. (2017). Building better open-source tools to support fairness in automated scoring. In *Proceedings of Ethics in natural language processing, Valencia, Spain* (p. 41) https://acl-arc.comp.nus.edu.sg//~antho/W/W17/W17-16.pdf#page=53

Mislevy, R. J., Almond, R. G., & Lukas, J. F. (2003). A brief introduction to evidence-centered design. *ETS Research Report Series, 2003*(1), i–29.

Mislevy, R. J., Corrigan, S., Oranje, A., Dicerbo, K., John, M., Bauer, M. I., Hoffman, E., von Davier, A. A., & Hao, J. (2014). *Psychometrics and game-based assessments*. Institute of Play.

Reichenberg. (2018). Dynamic Bayesian networks in educational measurement: Reviewing and advancing the state of the field. *Applied Measurement in Education, 31*, 335–350.

Ryans, D. G., & Frederiksen, N. (1951). Performance tests of educational achievement. In E. F. Lindquist (Ed.), *Educational measurement* (pp. 455–494). American Council of Education.

Sao Pedro, M. A., Baker, R. S., & Gobert, J. D. (2012). Improving construct validity yields better models of systematic inquiry, even with less information. In J. Masthoff, B. Mobasher, M. Desmarais, & R. Nkambou (Eds.), *International conference on user modeling, adaptation, and personalization* (pp. 249–260). Springer.

Shute, V. J., & Ventura, M. (2013). *Stealth assessment: Measuring and supporting learning in video games*. MIT Press.

Shu, Z., Bergner, Y., Zhu, M., Hao, J., & von Davier, A. A. (2017). An item response theory analysis of problem-solving processes in scenario-based tasks. *Psychological Test and Assessment Modelling, 59*(1), 109–131. https://www.psychologie-aktuell.com/fileadmin/download/ptam/1-2017_20170323/07_Shu.pdf

Stoeffler, K., Rosen, Y., Bolsinova, M., & von Davier, A. A. (2018). Gamified assessment of collaborative skills with chatbots. In C. P. Rosé et al. (Eds.), *Artificial Intelligence in Education. AIED 2018* (Lecture Notes in Computer Science) (Vol. 10948). Springer. https://doi.org/10.1007/978-3-319-93846-2_64

von Davier, A. A. (2015). Virtual and collaborative assessments: Examples, implications, and challenges for educational measurement. In F. Bach & D. Blei (Eds.), *Workshop on machine learning for education, international conference on machine learning*.

von Davier, A. A. (2017). Computational psychometrics in support of collaborative educational assessments. *Journal of Educational Measurement, 54*, 3–11. https://doi.org/10.1111/jedm.12129

von Davier, A. A., Wong, P., Polyak, S. T., & Yudelson, M. (2019a). The argument for a "Data Cube" for large-scale psychometric data. *Frontiers in Education*. https://doi.org/10.3389/feduc.2019.00071

von Davier, A. A., Deonovic, B., Yudelson, M., Polyak, S., & Woo, A. (2019b). Computational psychometrics approach for holistic learning and assessment systems. *Frontiers in Education*. https://www.frontiersin.org/articles/10.3389/feduc.2019.00069/full

Chapter 4
Virtual Performance-Based Assessments

Jessica Andrews-Todd, Robert J. Mislevy, Michelle LaMar, and Sebastiaan de Klerk

Abstract Virtual performance-based assessments (VPBAs) are environments for test takers to interact with systems, sometimes including other persons or agents, in order to provide evidence about their knowledge, skills, or other attributes. Examples include tasks based on interactive simulations, games, branching scenarios, and collaboration among students communicating through digital chats. They may be used for summative purposes, as in certification examinations, or for other purposes, as in intelligent tutoring systems and exploratory learning environments. They afford opportunities to obtain direct evidence about capabilities that inherently involve interaction, such as inquiry and collaboration. Our focus here is digital, usually with regard to the environment but always with regard to the form of data. Digital data capture makes it possible to acquire rich details about students' actions and the evolving situations in which they occur. The challenges they pose to psychometrics lie in designing VPBAs to optimally evoke the targeted capabilities, providing students with affordances that evidence that cognition, capturing the relevant aspects of the performances, identifying meaningful patterns in performances that constitute evidence about the targeted capabilities, and providing an inferential framework for synthesizing the evidence and characterizing its properties. This chapter provides an introduction to VPBAs and psychometric considerations in VPBA design and analysis.

The R or Python codes can be found at the GitHub repository of this book: https://github.com/jgbrainstorm/computational_psychometrics

J. Andrews-Todd (✉) · R. J. Mislevy · M. LaMar
Educational Testing Service, Princeton, NJ, USA
e-mail: jandrewstodd@ets.org; rmislevy@umd.edu; mlamar@ets.org

S. de Klerk
eX:plain/University of Twente, Enschede, The Netherlands
e-mail: s.deklerk@utwente.nl

© Springer Nature Switzerland AG 2021

A. A. von Davier et al. (eds.), *Computational Psychometrics: New Methodologies for a New Generation of Digital Learning and Assessment*, Methodology of Educational Measurement and Assessment, https://doi.org/10.1007/978-3-030-74394-9_4

4.1 The What, Why, and How of Virtual Performance-Based Assessment

In order to ground the more technical discussions of computational psychometric methods in the following chapters, we present in this chapter an overview of the virtual performance-based assessments (VPBAs) in which many of these methods are needed. We draw, in turn, on the discussions of sociocognitive psychological perspective, assessment argumentation, and psychometric principles of validity, reliability, generalizability, comparability, and fairness as the foundations on which psychometrics must extend to these new forms of assessment (Chap. 2).

Broadly speaking, the *what* of VPBA is providing an environment for test takers to interact with simulated situations that present important features and responsive behaviors in some domain, and give test takers affordances that simulate important features of the actions they would take, all to the end of providing evidence about their capabilities for thinking and acting in such situations. Some VPBAs engage a single test taker; some engage multiple test takers, as in an assessment of collaborative problem solving; others involve a test taker and other agents who are involved in the interaction but not being assessed, such as actors in a role-playing task or virtual non-player characters in a video game. Our focus in this chapter is VPBAs in which the environment is digital, although comparisons with real-life simulations such as role-playing tasks and troubleshooting actual mechanical systems allow us to draw on insights from a century of experience with performance assessments (e.g., Ryans & Frederiksen, 1951).

The *why* of VPBAs is the evidence they can provide about aspects of capability that are hard to assess directly with static assessment situations and final products. Familiar task types of this sort can indeed evoke at least some of the targeted knowledge and skills; for example, a description or a video clip of a problem-solving episode followed by multiple-choice questions about likely causes and next steps may well lead a test taker to construct and run a mental model of the situation – but we get no direct evidence about the construction, unprompted action choices, or interpretations and subsequent reactions the test taker might have taken. We get no direct evidence of the moment-by-moment actions a person takes to understand, act, react, and create the situations that constitute successful performance in a domain, which draw on knowledge and skills, to be sure, but on higher-order understandings and metacognition as well. Psycholinguist Lawrence Barsalou (1999) described comprehension as "preparation for situated action" (p. 61). So let's see some action, and learn what it can tell us about capabilities we are actually interested in!

The *how* is twofold. The first is psychometrics viewed narrowly as analytic procedures for modeling data from educational and psychological assessments. The second is a more extensive view, not limited to analyzing given data but encompassing the issues of design, purpose, theory, context, and content that lead to the data, and together establish the situated meanings of the data, the model parameters, and subsequent inferences through the model (Chap. 2). Because VPBAs are new and complicated, it may well be that the psychometric challenges

of the second kind are more demanding than those of the first kind. It is certain, however, that one cannot count on psychometric models or analytic procedures to save the day if there are serious failures in theory, design, or contextualization once data arrive on the psychometric doorstep. For this reason, the present chapter addresses these challenges as well as psychometric models *per se.*

In Sects. 4.2, 4.3, 4.4 and 4.5, then, we discuss central features, and resulting psychometric challenges, in sometimes-overlapping categories of VPBAs. They are scenario-based assessments, simulation-based assessments, game-based assessments, and assessments involving collaboration. Section 4.6 then provides a brief overview of psychometric methods that have been applied in VPBAs.

There are however certain observations based on the discussions in Chap. 2 that we mention here first, as they hold implications for psychometric methods in the kinds of VPBA we address:

1. Because most VPBAs are interactive, a test taker is interacting in what amounts to an evolving continuum of situations. A resulting feature of the argument structure is suggested by Fig. 4.1, though not as discrete or simply as the figure suggests. The figure suggests multiple, sequential, dependent copies of the basic assessment design/interpretation argument shown in Chap. 2 (both the basic structure and its extension to VPBAs are discussed further in Mislevy, 2018). Although certain situations and work products may be predetermined, both features of situations and features of performance with meanings that depend on situation features emerge from the interactions. Processes concerned with characterizing evidence from given performances are called *evidence identification* (in the figure, this is the arrow from the student's actions to "data concerning student performance"). The neat *a priori* partitioning of situations and methods of evidence identification that characterize discrete item assessment does not hold.
2. The data captured in work products are generally low-level, and often more than one level of evidence identification and aggregation is required. The evidence identification processes that hold evidence about targeted psychological capabilities are apt to be patterns across low-level actions, interpreted in light of the evolving situations in which they are produced (Chaps. 9, 10, 11, 12, 13 and 14).
3. A particular feature of the low-level actions in VPBAs is *conditional dependence*; that is, the evidence that a given action holds about a higher-level construct can depend on not only the current situation but perhaps previous situations, previous actions, and even subsequent actions when a given action could be a good first step of a viable strategy but is unproductive unless appropriate actions follow. While discrete-item assessments avoid conditional dependence, capitalizing on the relations among actions can be a valuable source of information in VPBAs (Chaps. 9, 10, 11, 12, 13 and 14).
4. In many cases, an assessment consists of multiple tasks, or test takers performing in a complex task can work themselves into qualitatively different situations, but it is still desired to draw inferences about higher-level overarching constructs.

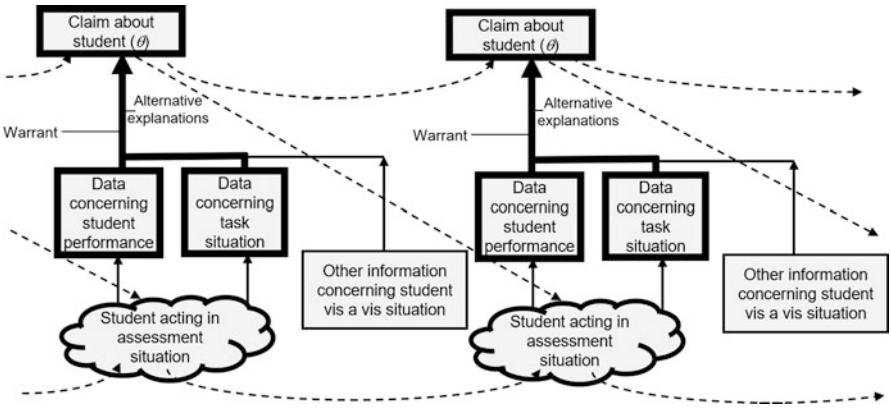


Fig. 4.1 Simplified design/interpretation argument in interactive task performances. Bold lines indicate the main elements that enter as variables in measurement models. Dashed lines indicate potential dependence over time, which may be to evidence-identification from performances and to inferences about students

Latent-variable psychometric models (Chap. 6) can be used in these cases to synthesize evidence in possibly different amounts, from different sources, or conveying evidence for different combinations of constructs. This is a strategy of evidence accumulation by means of a "psychometric backbone," as suggested in the heavy lines in Fig. 4.1. A static latent variable presumes that the corresponding capability can be approximated as being constant over the time period being modeled. A dynamic latent variable allows for the capability to change through the experience, as would be desirable in a VPBA meant to foster learning.

5. Multiple aspects of capability, characterized in terms of constructs, may be involved in a VPBA task. A multivariate latent-variable model is used if they need to be distinguished in inferences. Both designed-in and emergent features of a situation influence which constructs are modeled as being involved, in what combinations. Such models can be assembled in real time from preconstructed model components in accordance with the features of the evolving situations (Chap. 6).

4.2 Scenario-Based Assessments

A scenario-based assessment introduces a narrative context for a student's experience, to add a layer of meaning to motivations, thoughts, and actions. The simplest version introduces an ongoing context to a sequence of tasks, which are frequently in a traditional format (e.g. multiple-choice or short answer). More complex scenario-based assessments can add elements of simulation and gaming as discussed in

Fig. 4.2 Screenshot of the weather task

subsequent sections. The context provides motivation in the form of a narrative, but it also provides an opportunity to evaluate higher-level capabilities such as comparison of arguments or synthesis of preceding information and results.

For example, in science assessment it is common to motivate an assessment with a real-world problem in the context of which students are asked to solve discrete items. In one such assessment, called the Weather Task (See Fig. 4.2; Liu et al., 2016), students are asked to predict the likelihood of a thunderstorm. After an instructional animation, students are presented with a number of items that assess their understanding of the formation of thunderstorms. Throughout the assessment, students are presented with a variety of item types (e.g., constructed-response, selected-response, menu-based branching conversations) to assess their knowledge of earth science content and science inquiry. Frequently these items build on each other to solve the larger problem presented in the scenario.

Scenario-based assessments present an immediate challenge to psychometrics in their conditional dependence. Both the existence of a narrative and the fact that items build on each other break the traditional assumption of conditional independence. Differing responses on earlier items can result in a different set of tools and resources brought to later items. Additionally, the phrasing of later items can, intentionally or not, cue the student to the correctness of their previous responses. This allows students to learn during the assessment. Further, the lack of conditional independence prevents the use of simpler methods appropriate for conventional tests.

Scenario-based assessments also introduce a threat to generalizability. Situating a large part of the assessment within a single context prevents disambiguation

between context-specific effects and construct effects. As noted in Chap. 2, early learning is often context dependent, implying that particularly at the mid-ability level context could introduce a large amount of construct-irrelevant variance. To optimally utilize information from such assessments may require either collecting data from multiple assessments with differing contexts or analyzing subskills employed during the assessment to separate process capability from context-specific knowledge that may influence the formation of a final response.

Despite the psychometric difficulties introduced by the conditional dependence and generalizability concerns introduced by VPBAs, they also present an opportunity to measure constructs of higher-order thinking. Evidence of such higher-order thinking is contained in not only the final response in the sequence, but in the pattern of responses throughout the scenario. Aggregating such evidence together in a principled psychometric inference is still a matter of ongoing research; an overview of concepts and approaches are discussed in Sect. 4.6 below and addressed in in Part II of the present volume.

4.3 Simulation-Based Assessments

"*In simulation-based assessments, the goal is to create a particular situation that resembles the real-world context (e.g., conducting a science experiment, administering medical care, or flying a jet fighter) to provide opportunities to assess individuals' capabilities in such a situation*" (Davey et al., 2015). Simulation-based assessments are particularly important to afford opportunities for people to demonstrate their knowledge, skills, and abilities in situations that can be too expensive, too time consuming, or too dangerous to carry out in real-world contexts (Snir et al., 1993). It is important to note that simulation-based assessments are not themselves new, as they were used in instances in research dating back decades (Frederiksen, 1966). What is new is technology for authentic interaction and automated scoring that open the paradigm to many previously infeasible purposes, contexts, and scales.

For example, in the Weather Task, students can be given access to simulated weather stations and asked to run experiments to gather data. The student must decide how many stations to place, where to place them, and how long to collect data. One could imagine this simulation including the ability to travel to different places in the world that have different atmospheric conditions to determine what effect the varying conditions have on storm formation. A sufficiently complex simulation might comprise the whole assessment: a student would propose a series of hypotheses about the relationships among variables, propose experiments to test them, analyze the results, and state conclusions.

Compared with familiar forms such as multiple-choice items, simulation-based tasks can be both costly to develop and time-consuming to deliver. One way to deal with the issue of cost in developing multiple types of simulations is to develop parallel tasks which can amortize the investment into a class of simulations over

many parallel versions. However, the final result of the performance in simulations (such as the patient's condition at the end of treatment), whether dichotomized as correct/incorrect or rated on a scale, may not provide enough reliable evidence of capability to justify the simulation's use. Fortunately, the final result is not the only evidence available. Digital simulations provide a performance trace that can include actions the student took, captured at various levels of granularity, along with moment by moment changes to the simulated environment. Each event includes a timestamp, making action ordering and duration of pauses between actions available for analysis. These data are increasingly of interest to computational psychometrics as evidence of student thinking, problem-solving processes, and even motivation. It is worth noting that a traditional assessment item can be designed to require the cognitive processes of complex problem solving, but such items are unable to capture the processes that students use to solve the problem.

While the simulation-based assessments provide a rich source of evidence for student processes, research into methodologies for translating this evidence into inferences about students' capabilities is still in its early stages. Evidence identification becomes a non-trivial problem. Behaviors we are interested in are frequently recorded as a series of low-level interface actions, few of them meaningful in and of themselves. And while the log files can record in great detail what students actually *did* in terms of keystrokes and clicks, they cannot definitively tell us what students were *intending* to do. From the moment we associate a pattern of clicks with a behavior, we are making an inference under uncertainty. Additional complications include multiple paths to acceptable solutions, temporal dependence of behaviors, and differing contextual value of the same behaviors.

4.4 Game-Based Assessments

Game-based assessments incorporate principles of game design in addition to assessment design to measure individuals' knowledge, skills, and abilities. These types of assessment can increase motivation by making the tasks more engaging. In designing a game-based assessment, it is important to find the balance between design considerations for game elements and assessment elements. Maximizing the effectiveness of game-based assessment requires particularly the balance between ensuring there are game-like characteristics such as fun and engagement while also ensuring the game satisfies psychometric requirements such as validity, reliability, and fairness as needed to serve its intended purpose, and allows individuals to provide evidence of their capabilities (Kim & Shute, 2015). Given that game designers build complex, interactive experiences, game-based assessments are particularly suitable for complex constructs such as systems thinking and cross-cultural competence that have been increasingly deemed critical for success in many academic, workplace, and military contexts. As such, game-based assessments can allow us to assess constructs that are difficult to assess with traditional multiple-choice assessments.

One could imagine the Weather Task set in a game context by casting the thunderstorms as threats to valued resources. To succeed in the game the student would need to predict where and when the storms would form and move their resources accordingly. The skills required in the simulation-based assessment are still needed to succeed, but in the game-based version the student additionally takes actions that advance goals within a narrative. A design goal would be to provide mechanics such that coming to understand and act through the targeted knowledge and practices is integrated with game play (Mislevy et al., 2014; Ke et al., 2019).

Psychometrically, game-based assessments frequently combine all of the challenges of scenario-based and simulation-based assessment with new challenges that are unique to the framing of the assessment as a game. One kind of challenge concerns threats to validity such as construct irrelevant variance. Game-based assessments may bring about variance that is unrelated to the constructs one is trying to measure. For example, prior experience with game play can be a confounding variable when trying to make inferences about students based on their game performance. Given that game-based assessments have game elements, they also may not seem like "tests" to students and this can influence the way they perceive the assessment and how they behave during the assessment (DiCerbo et al., 2017). The game elements may evoke a game mindset that encourages unwanted experimentation or exploration rather than directed problem solving. Additional challenges with game-based assessments include potential individual differences with respect to things such as enjoyment of and desire to achieve a "top score" and that these types of assessment may allow students to make multiple attempts over a period of time which can create fewer items to score than in assessments that have discrete test items (Kerr et al., 2016).

4.5 Collaborative Assessments

Given that much of the planning, decision making, and problem solving in the modern world is carried out by groups, there is increasing attention in determining the extent to which individuals possess the skills necessary to succeed in group settings. This need has motivated the development of assessments that can measure collaborative skills in and of themselves rather than focusing solely on the measurement of content knowledge. *Collaborative assessments are designed to create opportunities for individuals to demonstrate their proficiency in working or communicating effectively with others.* Virtual environments are well-suited for this, as they can present complex problems and scenarios for individuals to work through in concert with others. In virtual environments, all the actions and discourse among team members can be captured rather than just the answers they provide. The resulting data can illuminate the processes and strategies team members use during performance, as to both the substance of the problem and the nature of the interactions.

Collaborative assessments can take different forms. They can include environments that afford communication and collaboration between one or more human participants. The communication may be asynchronous, such as with threaded online discussion in a learning management system or massive open online courses (MOOCs) (Rosé & Ferschke, 2016; Wise & Chiu, 2011), or synchronous in which individuals communicate in real time through a chat box or audio or video channels (Andrews-Todd & Forsyth, 2020; Andrews-Todd et al., 2019; Bower et al., 2017; van Joolingen et al., 2005). Collaborative assessments can also include environments that support communication between a human participant and one or more computer agents (Biswas et al., 2010; Graesser et al., 2012; OECD, 2013). There also exist environments that incorporate both collaborative designs and include two human participants and artificial agents (e.g., Liu et al., 2015).

To make the Weather Task collaborative, the problem would need to be modified to introduce enough complexity to necessitate communication and collaboration between at least two people. The task would also need to include a communication channel such as a text chat box (Andrews-Todd & Forsyth, 2020). One way to make the task require collaboration is to distribute resources needed to solve the problem among team members. For example, each team member can have access to two weather stations so that they must communicate to negotiate where to place the stations, how many stations to place, and how to interpret the data that are gathered from the stations.

The addition of collaborative features in assessments present challenges for measurement of individuals' performance. First, the contributions of individuals are not only dependent upon their own ability, but also the abilities and contributions of others with whom they are collaborating. This creates not only an issue of statistical dependence, but also one of standardization. A task, as well as the collaboration itself, may be easier or more difficult depending upon the skills and attributes of the collaborators. Methods that can deal with such dependencies are required.

Given the process data from collaborative tasks, evidence of collaborative skills can be gathered from the communication channel used by the team. Evidence identification is thus frequently faced with multimodal data, with the potential need to automatically detect features from video, audio, and/or text chat streams (e.g., Chap. 11). After identification, a psychometric model that incorporates the dynamic interaction among team members will be required. Further, collaborative skills are often multidimensional in nature. As a result, common psychometric methods such as classical test theory and unidimensional item response theory can prove insufficient for collaborative assessments.

4.6 Psychometric Methods in Virtual Performance-Based Assessments

The four preceding sections all ended with the psychometric challenges associated with each type of assessment. Scenario-based assessments are characterized by conditional dependence and generalizability concerns. There is a need for further research on psychometric inferences based on an aggregation of response patterns throughout the scenario. This also holds for simulation-based assessments, in which informative elements of evidence are often 'hidden' in the log file and need to be uncovered through exploratory and confirmatory psychometric methodologies. Game-based assessments combine the previous challenges with new 'game behavior validity challenges' that can induce construct irrelevant variance (e.g., prior experience, motivation, or an explorative mindset). Collaborative assessments are characterized foremost by the challenges of interpersonal dependencies and standardization. That is, the performance of an individual also depends on how others in the assessment are progressing and how the joint interactions unfold. In addition, collaborative assessments' multimodal and multidimensional data provide difficulties for psychometric modeling. From a methodological perspective, the different types of VPBA's can offer more standardization, strongly reduced rater error, and increased sampling of complex tasks (De Klerk et al., 2014). However, the full potential can only be realized if the development of the underlying 'psychometric machinery' of VPBA's keeps pace with the fast evolution of digital technology in education (DiCerbo, 2017). In this section, we therefore address the challenges discussed in the chapter through surveying the status of psychometric methods in VPBA.

Evidence Modeling The point of departure for this exploration is the *evidence model* within the *conceptual assessment framework* (CAF) layer of the *evidence-centered design* (ECD) framework (Mislevy et al., 2003). While it is beyond the scope of this chapter to discuss ECD or the CAF layer in depth, essentially the evidence model is where assessment tasks and students meet, and it is the home of the statistical and psychometric methodology. The following models embody, in analytic and psychometric models, the evidentiary-argument steps outlined in Chap. 2.

As students move through tasks in a VPBA, exhibiting all sorts of behavior, depending on the sophistication of the VPBA, they provide evidence about their knowledge, skills, and abilities (KSAs). From an evidence perspective it is necessary to determine which behavioral elements, the evidence in the VPBA, are indicative of the KSAs it is intended to measure. With regard to simulation-based assessments and game-based assessments, these evidentiary elements are often hidden in extensive log files. The process of locating and determining evidentiary elements is often called *evidence identification* (Levy, 2013). When these elements, often called Observable Variables (OVs), are determined, they can serve as input for a psychometric model. A psychometric model transforms OV scores (i.e., the

extent to which students were able to demonstrate the behavioral elements that were identified as evidence) into a unidimensional or multidimensional score or profile that represents belief about the values of a student's KSAs. The process of combining the identified evidentiary elements, which may take place in multiple stages and combine information from different kinds of data, is often referred to as *evidence accumulation* (Levy, 2013). Aggregating response patterns throughout scenario-based assessments is one of the challenges that should be addressed through evidence accumulation. In the following two paragraphs we discuss the status and recent psychometric developments within evidence identification and evidence accumulation in VPBAs.

Evidence Identification The statistical methodology that can be used for the process of evidence identification is both exploratory and confirmatory. Through multiple techniques the most informative behavioral elements in a VPBA regarding a student's KSAs may be identified. Many of these techniques are captured under the *educational data mining* (EDM) or *machine learning* (ML) paradigms (Chaps. 9, 10 and 11). EDM is concerned with finding meaningful relationships in big data that are logged in the VPBA (Rupp et al., 2012). Educational data mining techniques can, for example, be very useful for finding evidentiary elements in the log files of simulation-based assessments or game-based assessments. ML has the same objective but is a more general field of study, focusing more on computer science for finding these relationships (Witten et al., 2016). Such a system may be 'primed' with examples from which the computer can then "learn" how to progress to new data.

An example of EDM or ML is *cluster analysis* (Chap. 10). Cluster analysis is useful in finding clusters of data that are coherent in tasks that purportedly measure the same KSAs. It might also indicate whether conditional dependencies between tasks or elements of data exist, which is one of the challenges for scenario-based assessments as well as game-based assessments. Kerr and Chung (2012) applied cluster analysis to find meaningful relationships in the data provided by students who played the educational video game *Save Patch*. In this game, students learned concepts associated with sixth grade mathematics by placing trampolines in a one- or two-dimensional grid, then dragging coils on the trampolines that make it bouncy. The distance that Patch (the character that the student saves by solving fractions and adding numbers) can jump is based on the sum of all coil values added to the trampoline. Using cluster analysis, the researchers were able to identify specific solution strategies and error patterns, and 94% to 97% of students' actions in *Save Patch* appeared to have been accurately identified as belonging to the 55 clusters the researchers had identified.

An application of machine learning is *(artificial) neural network analysis* (ANN) (Chaps. 11 and 14). ANN systems can "learn" to do tasks, for example, and identify new problem-solving strategies. These networks can be programmed through several ways, but it is most common that a learning algorithm is used that is equipped to recognize whether a strategy is most optimal or has least cost. A famous example is AlphaGo, a computer program that plays the Chinese game Go,

a very complex board game. AlphaGo was able to defeat a professional Go player by means of its highly advanced ANN systems. In an educational context, Stevens et al. (2004) used a self-organizing ANN to identify the most common student problem-solving strategies in conducting online qualitative chemical analysis tasks. Using ANN and Hidden Markov Model Analysis (as discussed in Chaps. 10 and 13), they were able to identify multiple learning stages that could be used to predict whether students would follow a correct problem-solving strategy. These types of networks might be useful to answer psychometric challenges for the multimodal and multidimensional data that flow out of collaborative assessments. Potentially these networks can learn to discern strategies from different individuals within the assessment, which then can be scored. From a diagnostic point of view, such VPBA's may be helpful for improving students' learning trajectories. The examples discussed above only provide a small glimpse of how these methodologies can be applied, but they exemplify how EDM or ML can be used and which challenges they answer. In addition, these are only two of many other techniques discussed in the literature.

Evidence Accumulation The psychometric models used within the process of evidence accumulation are more confirmatory than exploratory. Logistically, the process of evidence accumulation follows after evidentiary elements have been identified in the evidence identification process, although it cannot be ruled out that both processes evolve simultaneously in a process of trial and error – for example, when the type(s) of evidence identified do not fit the psychometric model in mind. It should be noted further that the nature and grainsize of the variables in a higher-level psychometric model should be attuned to the purpose of the assessment. In fact, the same performances may be modeled at different grainsizes for the purposes of summary information to policy-makers and as diagnostic feedback to support students' learning (Oliveri et al., 2021).

A psychometric model that is often discussed in the context of VPBA is the *Bayesian network* (BN) (De Klerk et al., 2015). A BN is a graphical structure for reasoning under uncertainty (Chap. 6). For example, within a scenario-based assessment aggregation of evidence is needed to make sound psychometric inferences. A BN may be helpful for this process (Almond et al., 2015). As discussed in Chap. 5, they have been applied in adaptive learning systems as well.

BN's have also been applied to answer challenges for game-based assessment. They can for example also incorporate data on prior experiences with the game-based assessment or explorative behavior (e.g., Shute, 2011). The reasoning is based on a Bayesian psychometric modeling framework (Pearl, 1988). A network consists of nodes, observable and latent variables, which are connected through arcs which indicate that a conditional probabilistic relationship exists between two or more variables (Neapolitan, 2003). As already mentioned, OVs can be the evidentiary elements (i.e., the behavior of students in the VPBA), sometimes synthesized through multiple layers of calculation, while the latent variables are the higher-level KSAs that are intended to be measured. Students' actions in the VPBA then influence the state of belief about the KSAs through the conditional probabilities

that are identified in probability tables and update the network. In a base state, the probabilities may be defined by experts. The learning character of the network makes it possible to update probabilities based on empirical data. In that way, a BN can be improved continuously.

An example of how a BN has been used in an educational context is provided by Shute (2011). She had students perform several quests in a commercial video game called *Oblivion*. The BN was used to model and measure students' *creative problem solving* skills. For instance, students were required, among other quests, to bring a character to the other side of a river. The video game allows many ways to cross a river (e.g., find a bridge, swim, build a bridge, use a spell to freeze the river), and these were ranked, as with the other quests, by experts on novelty and efficiency. The conditional probabilities in the BN informed the researcher about the probability that a student was either low or high on creative problem solving skills. That is, each time students performed highly novel *and* efficient actions in the video game the probability that they are high on creative problem solving increases, whereas when they perform conservative and/or inefficient actions the probability that they are high decreases. The nature of the problems and potential solutions can vary considerably across quests, and the exploratory clustering from a database of actual student attempts provides customized yet algorithmic evidence identification for each student on each quest. All provide evidence for creative problem solving through the BN. This example reflects how BNs can flexibly handle unorthodox observable variables in game-based assessments.

Performance Data Analysis The discussion above shows that the focus in analyzing the performance data of students in VPBA is not only on the *product data* (i.e., the final outcomes of the assessment or the 'ones and zeros'), but also on the *process data*. Process data (Chap. 8), usually saved in log files, can be any student behavior inside the VPBA (e.g., navigational path, time spent on tasks, keystrokes, mouse clicks, etc.) and can be very useful in estimating students' KSAs (Rupp et al., 2012). The key is to identify and combine those data elements and trace them back to the KSA's intended to be measured by the VPBA. This may result in a complex structure wherein relationships exist between multiple OVs and KSAs. That is, student actions in the VPBA may be dependent upon and interact with multiple other OVs and KSAs.

With digital technology and statistical and psychometric models improving, we expect to see increasing possibilities to analyze VPBA performance data, thereby supporting the widespread use of VPBA in the next decades. In addition to the illustrations used above to introduce the basic elements of VPBAs, other examples can be found in following chapters of this volume and in the assessment literature. For simulation- and game-based assessment, for example, see the edited volumes of Ifenthaler et al. (2012) and O'Neil et al. (2016). Collections of designs and computational-psychometric analyses for collaborative assessments can be found in the edited volume of von Davier et al. (2017) and the Spring 2017 special issue of the *Journal of Educational Measurement* on collaborative assessment. The latter

includes von Davier's (2017) article "Computational psychometrics in support of collaborative educational assessments."

4.7 Conclusions

VPBAs offer the potential to both expand and deepen the domain of assessment, and there is much enthusiasm about their use in and outside the field of education (Levy, 2013; Mislevy et al., 2014). With the digital revolution accelerating at an unprecedented pace, the growth of technological possibilities in assessment seems to be endless (Collins & Halverson, 2018). Now may be the time to capitalize on the full potential that VPBA's provide.

Critical challenges now faced with VPBAs, however, lie in integrating their psychological grounding, design, and analysis to best meet their potential. While VPBAs offer great potential, research on these evidentiary issues is in early stages. Although they first appeared and took their initial forms with simpler tasks and simpler psychological perspectives, the concepts and methods from psychometrics can play a critical role in developing a sound methodology for these new forms of assessment–hence this volume on computational psychometrics. In this chapter we have sketched the key features of some of the new kinds of assessment environments, and begun to connect them to psychometric concepts and methods. The chapters that follow provide more detail on techniques that are beginning to show their value in the new generation of assessments.

References

Almond, R. G., Mislevy, R. J., Steinberg, L. S., Williamson, D. M., & Yan, D. (2015). *Bayesian networks in educational assessment*. Springer.

Andrews-Todd, J., & Forsyth, C. M. (2020). Exploring social and cognitive dimensions of collaborative problem solving in an open online simulation-based task. *Computers in Human Behavior, 104*, 105759. https://doi.org/10.1016/j.chb.2018.10.025

Andrews-Todd, J., Jackson, G. T., & Kurzum, C. (2019). *Collaborative problem solving assessment in an online mathematics task (Research Report No. RR-19-24)*. Princeton, NJ: Educational Testing Service. https://doi.org/10.1002/ets2.12260

Barsalou, L. W. (1999). Language comprehension: Archival memory or preparation for situated action? *Discourse Processes, 28*, 61–80.

Biswas, G., Jeong, H., Kinnebrew, J. S., Sulcer, B., & Roscoe, R. (2010). Measuring self-regulated learning skills through social interactions in a teachable agent environment. *Research and Practice in Technology Enhanced Learning, 5*, 123–152.

Bower, M., Lee, M. J., & Dalgarno, B. (2017). Collaborative learning across physical and virtual worlds: Factors supporting and constraining learners in a blended reality environment. *British Journal of Educational Technology, 48*(2), 407–430.

Collins, A., & Halverson, R. (2018). *Rethinking education in the age of technology: The digital revolution and schooling in America*. Teachers College Press.

Davey, T., Ferrara, S., Shavelson, R., Holland, P., Webb, N., & Wise, L. (2015). *Psychometric considerations for the next generation of performance assessment*. Educational Testing Service.

De Klerk, S., Eggen, T. J. H. M., & Veldkamp, B. P. (2014). A blending of computer-based assessment and performance-based assessment: Multimedia-Based Performance Assessment (MBPA). The introduction of a new method of assessment in Dutch Vocational Education and Training (VET). *Cadmo, 22*(1), 39–56. https://doi.org/10.3280/CAD2014-001006

De Klerk, S., Veldkamp, B. P., & Eggen, T. J. H. M. (2015). Psychometric analysis of the performance data of simulation-based assessment: A systematic review and a Bayesian network example. *Computers & Education, 85*, 23–34.

DiCerbo, K. E. (2017). Building the evidentiary argument in game-based assessment. *Journal of Applied Testing Technology, 18*(1), 7–18.

DiCerbo, K. E., Shute, V., & Kim, Y. J. (2017). The future of assessment in technology-rich environments: Psychometric considerations. In J. M. Spector, B. Lockee, & M. Childress (Eds.), *Learning, design, and technology: An international compendium of theory, research, practice, and policy* (pp. 1–21). Springer.

Frederiksen, N. (1966). Validation of a simulation technique. *Organizational Behavior and Human Performance, 1*(1), 87–109.

Graesser, A. C., D'Mello, S., Hu, X., Cai, Z., Olney, A., & Morgan, B. (2012). AutoTutor. In P. McCarthy & C. Boonthum-Denecke (Eds.), *Applied natural language processing: Identification, investigation and resolution* (pp. 169–187). IGI Global.

Ifenthaler, D., Eseryel, D., & Ge, X. (Eds.). (2012). *Assessment in game-based learning*. Springer.

Ke, F., Shute, V., Clark, K. M., & Erlebacher, G. (2019). *Interdisciplinary design of game-based learning platforms*. Springer.

Kerr, D., Andrews, J. J., & Mislevy, R. J. (2016). The in-task assessment framework for behavioral data. In A. A. Rupp & J. P. Leighton (Eds.), *The handbook of cognition and assessment: Frameworks, methodologies, and applications* (pp. 472–507). Wiley-Blackwell.

Kerr, D., & Chung, G. K. W. K. (2012). Identifying key features of student performance in educational video games and simulations through cluster analysis. *Journal of Educational Data Mining, 4*(1).

Kim, Y. J., & Shute, V. J. (2015). The interplay of game elements with psychometric qualities, learning, and enjoyment in game-based assessment. *Computers & Education, 87*, 340–356.

Levy, R. (2013). Psychometric and evidentiary advances, opportunities, and challenges for simulation-based assessment. *Educational Assessment, 18*(3), 182–207. https://doi.org/10.1080/10627197.2013.814517

Liu, L., Steinberg, J., Qureshi, F., Bejar, I., & Yan, F. (2016). Conversation-based assessments: An innovative approach to measure scientific reasoning. *Bulletin of the IEEE Technical Committee on Learning Technology, 18*(1), 10–13.

Liu, L., von Davier, A. A., Hao, J., Kyllonen, P., & Zapata-Rivera, J.-D. (2015). A tough nut to crack: Measuring collaborative problem solving. In Y. Rosen, S. Ferrara, & M. Mosharraf (Eds.), *Handbook of research on computational tools for real-world skill development* (pp. 344–359). IGI-Global.

Mislevy, R. J. (2018). *Sociocognitive foundations of educational measurement*. Routledge.

Mislevy, R. J., Oranje, A., Bauer, M., von Davier, A. A., Hao, J., Corrigan, S., Hoffman, E., DiCerbo, K., & John, M. (2014). *Psychometric considerations in game-based assessment*. Institute of Play.

Mislevy, R. J., Steinberg, L. S., & Almond, R. A. (2003). On the structure of educational assessments. *Measurement: Interdisciplinary Research and Perspectives, 1*, 3–67.

Neapolitan, R. E. (2003). *Learning Bayesian networks*. Prentice-Hall.

OECD. (2013). *PISA 2015 collaborative problem solving framework*. OECD Publishing.

Oliveri, M. E., Mislevy, R., & Elliot, N. (2021). Principled development of workplace English communication, Part 1: A sociocognitive framework. *Journal of Writing Analytics, 5*, 34–70.

O'Neil, H. F., Baker, E. L., & Perez, R. S. (Eds.). (2016). *Using games and simulations for teaching and assessment: Key issues*. Routledge.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. Morgan Kaufmann.

Rosé, C. P., & Ferschke, O. (2016). Technology support for discussion based learning: From computer supported collaborative learning to the future of massive open online courses. *International Journal of Artificial Intelligence in Education, 26*(2), 660–678.

Rupp, A. A., Levy, R., DiCerbo, K., Sweet, S. J., Crawford, A. V., Calico, T., Benson, M., Fay, D., Kunze, K. L., Mislevy, R. J., & Behrens, J. T. (2012). Putting ECD into practice: The interplay of theory and data in evidence models within a digital learning environment. *Journal of Educational Data Mining, 4*(1), 49–110.

Ryans, D. G., & Frederiksen, N. (1951). Performance tests of educational achievement. In E. F. Lindquist (Ed.), *Educational measurement* (pp. 455–494). American Council of Education.

Shute, V. J. (2011). Stealth assessment in computer-based games to support learning. In S. Tobias & J. D. Fletcher (Eds.), *Computer games and instruction* (pp. 503–523). Information Age Publishing.

Snir, J., Smith, C., & Grosslight, L. (1993). Conceptually enhanced simulations: A computer tool for science teaching. *Journal of Science Education and Technology, 2*(2), 373–388.

Stevens, R., Soller, A., Cooper, M., & Sprang, M. (2004). Modeling the development of problem solving skills in chemistry with a web-based tutor. In *Proceedings of the 7th international conference on intelligent tutoring systems* (pp. 580–591).

van Joolingen, W. R., de Jong, T., Lazonder, A. W., Savelsbergh, E. R., & Manlove, S. (2005). Co-Lab: Research and development of an online learning environment for collaborative scientific discovery learning. *Computers in Human Behavior, 21*, 671–688.

von Davier, A. A. (2017). Computational psychometrics in support of collaborative educational assessments. *Journal of Educational Measurement, 54*(1), 3–11.

von Davier, A. A., Zhu, M., & Kyllonen, P. C. (Eds.). (2017). *Innovative assessment of collaboration*. Springer.

Wise, A. F., & Chiu, M. M. (2011). Analyzing temporal patterns of knowledge construction in a role-based online discussion. *International Journal of Computer-Supported Collaborative Learning, 6*(3), 445–470.

Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2016). *Data mining: Practical machine learning tools and techniques*. Morgan Kaufmann.

Chapter 5
Knowledge Inference Models Used in Adaptive Learning

Maria Ofelia Z. San Pedro and Ryan S. Baker

Abstract This chapter provides an overview of adaptive learning and examines the student model component used in adaptive learning systems. Established and more recent approaches to student modeling that infer student knowledge (i.e. what students know at any given moment during the learning experience) are discussed, as student knowledge is the most common learner characteristic widely assessed in large-scale adaptive systems. This chapter concludes with a discussion of the limitations of the current generation of adaptive learning systems, and areas of potential for future progress.

In recent years, digital environments for learning and assessment have come to provide rich opportunities to capture and utilize data on the learning experiences of students, increasing the potential to fully personalize learning in these environments. When online learning systems use data to tailor to student characteristics and performance, they become adaptive learning environments. Although existing systems generally fall short of this idea (see review in Baker, 2016), an ideal adaptive learning environment would adapt to assessments of a wide range of student attributes and constructs influential to the learning experience – student knowledge, academic emotions, behavioral engagement and motivation. Successful adaptive learning systems can utilize detailed learner activity data in real-time to make inferences which are in turn used to automatically provide the learner with individualized support. Such support can come in the form of *adaptive sequencing* in

The R or Python codes can be found at the GitHub repository of this book: https://github.com/jgbrainstorm/computational_psychometrics

M. O. Z. San Pedro (✉)
ACT, Inc., Iowa City, IA, USA

R. S. Baker
University of Pennsylvania, Philadelphia, PA, USA

© Springer Nature Switzerland AG 2021

A. A. von Davier et al. (eds.), *Computational Psychometrics: New Methodologies for a New Generation of Digital Learning and Assessment*, Methodology of Educational Measurement and Assessment, https://doi.org/10.1007/978-3-030-74394-9_5

the student's learning experience by automatically adjusting the sequence of the next content or skills to master (VanLehn (2006)'s "outer-loop"), including adjusting difficulty of the items given, and also in *adaptive content* in response to a student's unique response, such as corrective feedback, context-sensitive hints, supportive messages, metacognitive advice (VanLehn's "inner-loop")..

Examples of adaptive learning systems that have successfully emerged over the years include intelligent tutoring systems (ITSs), adaptive platforms developed by educational publishers, adaptive educational games, adaptive educational hypermedia systems, and adaptive online courseware. The combination of maturing technology and government educational initiatives has led to a growth in the number of adaptive learning systems used by schools. Increasingly, ITSs designed to teach math skills are being used at large-scale (hundreds of thousands of users each year, in schools and districts across the US), including Cognitive Tutors from Carnegie Learning, shown to lead to better performance on standardized exams (Pane et al., 2014), and subsequent math courses (Koedinger et al., 2000); and ALEKS from McGraw Hill Education, shown to increase class attendance and math performance when implemented in after-school programs (Craig et al., 2011). Some systems have been designed to provide a platform for instructor users to develop and import their content and pedagogical practices (e.g. Smart Sparrow, CogBooks, ASSISTments), while others provide instructional content created by the provider (e.g. Adapt Courseware, LearnSmart, Reasoning Mind).

The development of adaptive learning systems has in many cases leveraged methods from learning analytics (LA) and educational data mining (EDM) (Baker & Siemens, 2014), communities that employ machine learning and statistical techniques in the processing and modeling of fine-grained data. An alternate paradigm, Computational Psychometrics (CP), aims to integrate data-driven computer science methods (machine learning and data mining, in particular) in theoretical psychometric frameworks to support accurate and valid measurements of latent abilities in real time (Von Davier, 2017). Computational psychometrics attempts to address data dependencies and multidimensionality that are not easily handled by traditional psychometric models (e.g., IRT), using psychometric approaches towards handling the hierarchy in educational data. Both EDM/LA and CP have the goal of measuring learners' proficiencies, skills, and other attributes. A comprehensive discussion of the differences between EDM/LA and CP is, however, outside the scope of this chapter.

In this chapter, we provide an overview of adaptive learning and describe its essential components. In particular, this chapter will focus on several student modeling approaches that evaluate *what students know* (i.e. student knowledge) at any given moment during the learning experience, as this is the most common learner characteristic widely assessed in large-scale adaptive systems (Pelánek, 2017a). We then look into how these approaches are themselves validated, and discuss examples of how adaptive systems tailor learner experiences based on those assessments. We conclude with a discussion of the limitations of the current generation of adaptive learning systems, and areas of potential for future progress.

5.1 What Is Adaptive Learning?

Adaptive learning in educational settings is a method for personalizing learning experiences for students. Adaptive learning uses technology (typically online systems) and the data the system collects to evaluate students' interactions and performance towards identifying their needs and tailoring instruction, remediation, and other aspects of the learning experience to aid students to learn the material and succeed. The goal is to provide the best possible learning content and resources for the students' needs at a particular point in time. While most extant systems used at scale adapt to students' knowledge and limitations in understanding, research systems have adapted to other aspects as well, such as affect and motivation (Arroyo et al., 2014; D'Mello et al., 2010; Rebolledo-Mendez et al., 2005). Adaptive learning leverages findings and methods from cognitive science, educational psychology and artificial intelligence, in order to achieve these goals. Modern adaptive learning systems can identify what a student knows and does not know, their skills, and strategies, and can pin down the source of the misconceptions. They can in many other cases detect a student's learning goals, affective state, level of engagement, and their motivation to learn.

It is important to note that adaptive learning is different from adaptive assessment or testing. As mentioned earlier, adaptive testing estimates the student's proficiency to select the next test item to present to the student. The goal of adaptive testing is to maximize the amount of information about proficiency gained from the test while minimizing the amount of time required to complete the test. In contrast, the goal of adaptive learning is to optimize the student's learning experience by both assessing his or her proficiency or behavior and providing the necessary individualized support towards mastery and achievement. This is similar to the framework of Reinforcement Learning (RL), which models sequential decisions where an agent interacting with an environment learns a policy (a mapping of states to action) that maximizes a reward (Sutton & Barto, 1998). In the context of learning and instruction, these states can be students' cognitive states, the actions are instructional activities (e.g., problems, worked examples, video content, etc.) that can modify the student's cognitive state, and the reward can be a measure of the quality of the action (e.g., reward for a student learning a skill).

One core aspect of most adaptive learning systems is the explicit use of a student model (Sottilare et al., 2013; Wenger, 1987; Woolf, 2010) that consists of measures of the student's knowledge, goals, engagement, and other pertinent attributes that enable the system to make informed decisions in adapting to individual students. (Note that this use of the term "student model" is different from the use seen in the Evidence-Centered Design framework, i.e. Mislevy & Riconscente, 2006, and precedes it by a considerable amount of time). Student models can contain assessments of each of these constructs, and update them in real time, so that they can be used in adaptation or provided to the instructor. It is worthwhile to note that adaptive learning systems can be designed to use explicit student models first when deciding on a specific adaptation to a student (e.g., model-based RL) or only use

data to learn an adaptation without any student models (e.g., model-free RL), as we discuss later in this chapter.

5.2 Framework for Adaptive Learning Systems

At their core, online or digital learning systems are made adaptive by being built with three major components: a domain model, a student model, and a pedagogical or instructional model (Fig. 5.1). This is broadly the same set of components that has been used to describe intelligent tutoring systems (ITSs) (Sottilare et al., 2013; Wenger, 1987; Woolf, 2010), and similar to the components used to describe adaptive educational hypermedia (AEH) (Brusilovsky et al., 2004; Karampiperis & Sampson, 2005). The *domain model* refers to the content domain to be taught by the system, containing the set of topics and the corresponding learning objectives or outcomes, skills, knowledge, and strategies needed to learn them (Nkambou et al., 2010; Sottilare et al., 2013; Woolf, 2010). It is a representation of the ideal expert knowledge which can be in the form of content standards, prerequisite knowledge, or common misconceptions that students normally exhibit for a skill or content. The *student model* assesses the student users' cognitive state, as well as other

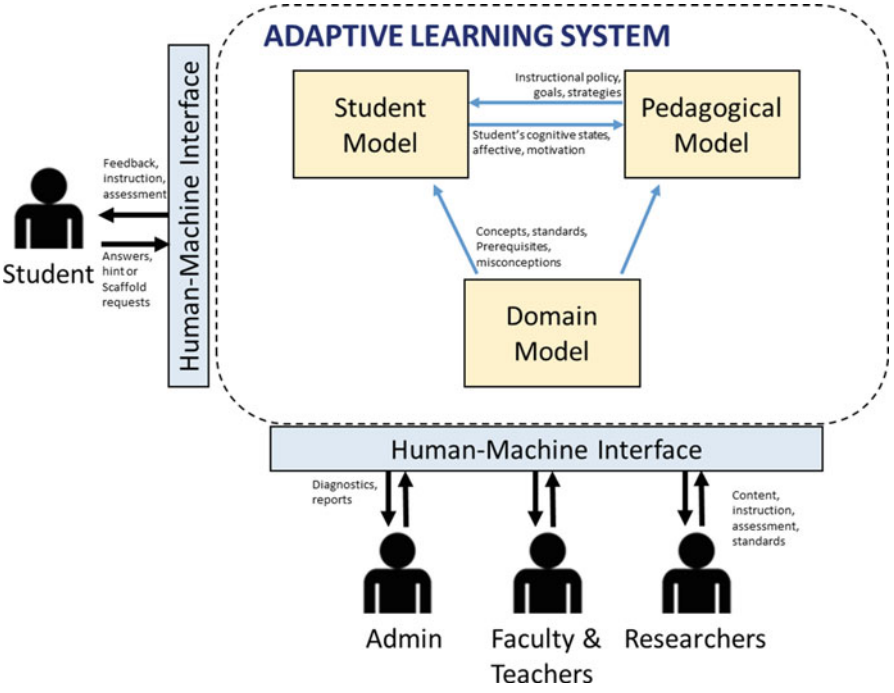


Fig. 5.1 Adaptive Learning Systems Framework

states (such as affective, behavioral and motivational states) based on the student's interaction with the learning system. The information from the student model is what adaptive learning systems consider in selecting the content and instruction to present to the student user. The *pedagogical or instructional model* selects the specific content, tutoring strategies and feedback for an individual student at a specific time (generally designed based on theories in pedagogy and instruction; in RL, it is referred algorithmically as policy or the instructional policy). It takes information from the domain and student models as inputs and decides the next activity the system will provide to the student user based on that information (Sottilare et al., 2013; VanLehn, 2006). While this component allows for automated adaptation in terms of deciding what to do next, certain mixed-initiative systems allow students to initiate the actions, ask questions or request for help (Aleven et al., 2006).

5.3 Adapting to Student Knowledge

Developing a good student model is usually seen as the starting point in the development of adaptive learning systems. There has been considerable research to determine which learner attributes can form the basis of meaningful and effective intervention, while at the same time being tractable to capture within a real-world system (Ahn & Brusilovsky, 2009; Biswas et al., 2005; Canfield, 2001; Graesser et al., 2007; Hu et al., 2012; Jeremic et al., 2009; Koedinger & Corbett, 2006; Murray & Arroyo, 2002; Razzaq et al., 2005; VanLehn, 2006). While there are several approaches to adaptability, most of current implemented adaptive learning systems are characterized by inferring student knowledge, and then adapting in an immediate fashion to that inference, for example by providing a hint to a struggling student or by providing mastery learning – giving the student additional practice on a difficult skill until he or she demonstrates mastery. Learning systems that adapt based on machine-learning based detection of student affect or meta-cognition, while successful in limited studies (Azevedo et al., 2012; D'Mello et al., 2009; DeFalco et al., 2018), have not yet scaled the way that knowledge-based adaptivity has. Systems using mastery learning represent student knowledge as skills, facts, or concepts, and track student progress from one problem or activity to the next, building a skill mastery profile for the students in order to eventually make decisions about when the student should move on to the next topic.

One core example of an adaptive learning system that uses knowledge estimates within mastery learning is the Cognitive Tutor (Koedinger & Corbett, 2006), mentioned above. The system makes use of a probabilistic user model that assesses student knowledge in real-time. With each student action (e.g., answering correctly or incorrectly, requesting for help) linked to a skill (or sometimes referred to as a knowledge component or KC) needed to solve a problem, the system evaluates these actions and aggregates these evaluations into evidence about which skills the student has learned or not learned (Corbett & Anderson, 1995), and displays this

skills progress to students and teachers. The system then provides the student with practice on the current skill until the student demonstrates mastery of that skill.

Andes is an adaptive learning system that teaches Newtonian physics using coached problem solving (VanLehn, 1996) where the system and the student collaborate in solving a problem. Its student model uses a Bayesian Network probabilistic framework that assesses a student's plans in solving the problem, predicts the student's goals and actions, and provides a long-term evaluation of a student's domain knowledge. The tutor updates its student model based on the problems the student has worked on and the student's fine-grained interactions with the system during problem solving, assessing the probability and uncertainty of the student's knowledge state (Conati et al., 1997).

Another example of an adaptive learning system that evaluates student knowledge in its student model is the SQL-Tutor, an intelligent tutoring system used in courses on database theory and the SQL database query language (Mitrovic, 1998). It selects a problem based on a constraint-based student model (Ohlsson, 1994) that represents correct knowledge in solving a problem as a set of constraints – aspects a correct solution must have. If a student's solution violates a constraint, the system intervenes through feedback attached to that constraint. This feedback informs the student what is incorrect, why it was incorrect and points out the corresponding domain principle attached to that constraint.

By modeling what a student knows, and updating a student's skill mastery profile in real-time, the system can then organize instruction around what the student knows, as well as provide feedback on the student's current state of knowledge. There have been several modeling approaches to infer what a student knows at a given time. In this section, we review one widely-used knowledge inference model, and then discuss other recent approaches to inferring student knowledge.

5.3.1 Bayesian Knowledge Tracing

To model and quantify student learning, many adaptive learning systems employ an approach known as Bayesian Knowledge Tracing (BKT) (Corbett & Anderson, 1995). BKT aims to infer the latent construct of the student's knowledge state for a skill (i.e. does a student know a certain skill at a given time?) from a student's pattern of correct and incorrect answers to problems or problem steps that require that skill up until that point. The model posits that at any given moment to demonstrate a skill, the knowledge state of a student is either mastered or not mastered, with a certain probability that is calculated given the student's observed performance of correct and incorrect responses (Fig. 5.2). It updates its knowledge inference that the student has mastered a skill or not on each opportunity of demonstrating that skill (e.g., through an item or problem step). Some extensions to Bayesian Knowledge Tracing have also added probabilistic correctness on items as well as binary correctness (right, wrong) (e.g. Sao Pedro et al., 2013).

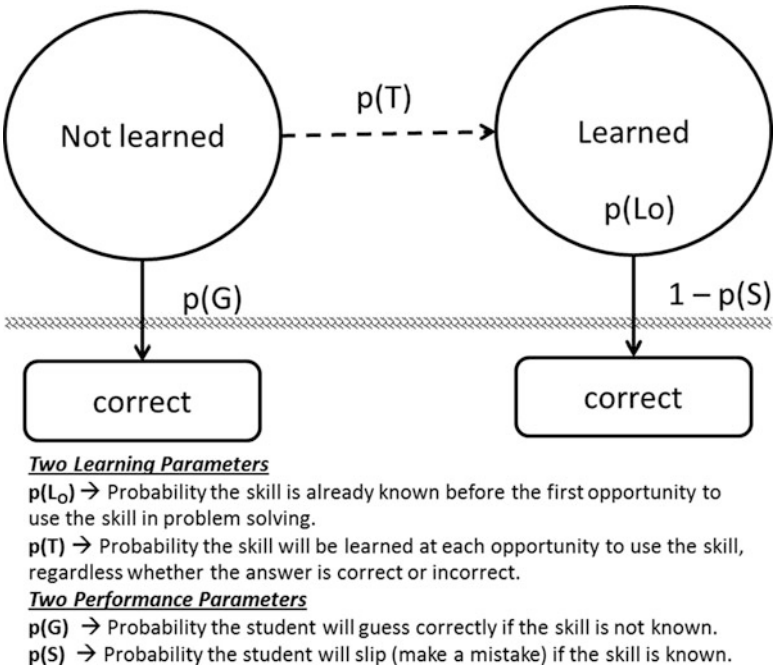


Fig. 5.2 Corbett and Anderson's model of Bayesian Knowledge Tracing

In its original formulation, only the first attempt at an opportunity to practice or demonstrate a skill (sometimes referred to as a practice opportunity) is considered, and each item or problem step corresponds to only a single skill. A set of four parameters is fit for each skill, and these four parameters remain invariant across the entire context of using the learning system: two learning parameters, $P(L_0)$ or initial knowledge (initial probability of knowing the skill) and P(T) or probability of learning the skill (at each opportunity to make use of the skill), together with two performance parameters P(G) or guess (probability that the student will give a correct answer despite not knowing a skill) and P(S) or slip (probability that the student will give an incorrect answer despite knowing the skill). Both the expectation maximization algorithm and grid search have been used to estimate the parameters from training data. The assumption is made that students do not forgot a skill once they have learned it. Another assumption is that all items, within the same skill, have the same difficulty, and that no contextual factors (beyond what skill it is) impact student performance or learning. Reye (2004) shows how BKT equations could be derived from a Hidden Markov Model or Dynamic Bayesian Network. Several algorithms have been used to estimate the parameters from training data, with approximately equal performance; several packages are available on the web to fit BKT parameters.

Using Bayesian analysis below, the system continually updates the estimate that a student knows a skill, whenever a student gives a first response to an item or problem step (correct or incorrect, bug, help request; help requests are treated as evidence that the student does not know the skill). This is done by first re-calculating the probability that the student knew the skill before making that first attempt (Eqs. 5.1 and 5.2), and then updating the probability based on the possibility that the student learned the skill at the current item or problem step (Eq. 5.3).

$$P\left(L_{n-1}|Correct_n\right) = \frac{P\left(L_{n-1}\right) * (1 - P(S))}{P\left(L_{n-1}\right) * (1 - P(S)) + (1 - P\left(L_{n-1}\right)) * P(G)} \tag{5.1}$$

$$P\left(L_{n-1}|Incorrect_n\right) = \frac{P\left(L_{n-1}\right) * P(S)}{P\left(L_{n-1}\right) * P(S) + (1 - P\left(L_{n-1}\right)) * (1 - P(G))} \tag{5.2}$$

$$P\left(L_n|Action_n\right) = P\left(L_{n-1}|Action_n\right) + ((1 - P\left(L_{n-1}|Action_n\right)) * P(T)) \tag{5.3}$$

Two common parameter fitting methods for standard BKT[1] uses Expectation-Maximization (EM)[2] and brute force grid search.[3]

One of the key assumptions of BKT is that its parameters vary by skill or knowledge components but remain constant for all other factors. In recent years, significant research have been conducted to remove or modify this assumption – either by relaxing one of its assumptions or considering additional information. For example, researchers have contextualized the parameter estimates of guess and slip using number of attempts and time spent (Baker et al., 2008), contextualized the four original parameters with the use of help (Beck et al., 2008), individualized the initial probability of mastery (prior per-student) (Pardos & Heffernan, 2010), added item-level parameters difficulty in BKT (González-Brenes et al., 2014; Khajah et al., 2014; Pardos & Heffernan, 2011), and have accounted for student differences in terms of initial mastery probabilities and skill learning probabilities (Yudelson et al., 2013).

[1]A scalable code written in C/C++ that fits Hidden Markov Models (BKT specifically) at scale can be found at https://github.com/myudelson/hmm-scalable

[2]EM used in modeling BKT with Bayes Net Toolbox for Student Modeling (BNT-SM). Matlab code can be found at http://www.cs.cmu.edu/listen/BNT-SM/

[3]Brute force grid search program code written in Java can be found at http://www.upenn.edu/learninganalytics/ryanbaker/edmtools.html

5.3.2 Performance Factor Analysis

Another relatively prominent model for student knowledge inference in ITSs is Performance Factor Analysis (PFA) (Pavlik et al., 2009). In PFA, the student's skill estimate is represented by estimating the probability that a student will get the item related to that skill correct. PFA employs a logistic regression model to predict student performance for a new item in terms of the number of success and failures (correctness or incorrectness) attained for the skills involved in that item so far. Unlike BKT, PFA assumes each item may involve multiple latent traits or skills. Each skill in the PFA model has the following parameters:

β the difficulty of the skill (can also represent item difficulty in some model variants)
γ the effect of prior successes tracked s for a skill (or KC), or success learning rate
ρ the effect of prior failures tracked f for a skill (or KC), or failure learning rate

Together with tracking the prior successes (s) and prior failures (f) for the KC for the student on each relevant skill, the probability that the learner will get the next item correct ($P(m)$)is computed (with Expectation Maximization used to fit the parameters):

$$m\left(i, j \in KCs, s, f\right) = \sum_{j \in KCs} \left(\beta_j + \gamma_j s_{i,j} + \rho_j f_{i,j}\right) \tag{5.4}$$

$$P(m) = \frac{1}{1 + e^{-m}} \tag{5.5}$$

In published comparisons, PFA has achieved predictive performance roughly comparable to BKT (see review in Pardos et al., 2011).

5.3.3 Deep Knowledge Tracing

Deep knowledge tracing (DKT), introduced by Piech et al. (2015), is based on deep learning or recurrent neural networks (RNN) (Rumelhart et al., 1985) to represent the latent knowledge state. Instead of producing a separate model for each skill, DKT models all skills together. The input to the RNNs consists of compressed representations of a student action (one-hot-encodings), resulting in a vector of predictions on the probability of getting items or exercises in the dataset correct. By using the sequence of student performance across skills, DKT provides predictions of a student's performance on future items within the system. Aside from the input and output layers, the RNN model also has a hidden layer with recurrent connections that serves to retain relevant aspects of the input that are useful for predicting future performance (Khajah et al., 2016). As a student progresses, DKT

utilizes information from previous items when useful to make inferences regarding future performance in every skill. RNNs have a high dimensional, continuous, representation of latent state giving them the ability to use this rich information as input in making prediction at a much later point in time. DKT has thus far been trained using stochastic gradient descent on mini batches, optimizing for negative log likelihood.

Khajah et al. (2016) found that modern extensions to BKT (inclusion of forgetting, latent student abilities and skill induction) achieved performance equivalent to DKT, though both approaches had higher predictive accuracy than Classical BKT. However, more recent extensions to DKT have proven better than BKT and PFA both at predicting future performance within the learning system (Yeung & Yeung, 2018; Zhang et al., 2017) and at predicting future performance on paper post-tests (Scruggs et al., 2020).

5.3.4 Elo Rating System

An emerging paradigm for predicting student knowledge in adapting learning systems is the Elo Rating System (Pelánek, 2016). Traditional Item Response Theory (IRT) assesses a student's knowledge of a particular topic by computing the probability that a student will get an item correct based on a fixed ability parameter for the student and difficulty and (sometimes) discriminability parameters of the item. It assumes there is only one skill being measured per set of items, though other psychometrics models relax this assumption (e.g., CDM, Rupp et al., 2010; multidimensional IRT, Embretson & Reise, 2000). IRT also assumes that no learning occurs between items, a problematic assumption in online adaptive learning, where a lack of learning would imply that the learning system was fundamentally failing in its goals. The Elo Rating System (Pelánek, 2016), a variant of the 1-PL IRT model, relaxes this assumption and can be used in a running system: the Elo Rating System (Pelánek, 2016) which is. The Elo Rating System continually estimates item difficulty and student ability, updating both every time a student encounters an item. It was originally devised for chess rating but has been used in student modeling by evaluating a student's item response as a "match" between the student and the item.

Applying the Elo rating system in the context of adaptive learning systems, Pelanek (2016) denotes the skill of a student s as θ_S, item (i) difficulty as d_i and the student response as $correct_{si} \in \{0, 1\}$. A logistic function using the skill and item difficulty gives the probability of a correct answer:

$$P\left(correct_{si} = 1\right) = \frac{1}{1 + e^{-(\theta_S - d_i)}} \tag{5.6}$$

$$\theta_S := \theta_S + K \bullet \left(correct_{si} - P\left(correct_{si} = 1\right)\right) \tag{5.7*}$$

$$d_i := d_i + K \bullet (P\,(correct_{si} = 1) - correct_{si} \qquad (5.8^*)$$

*where K is a parameter for how strongly the model should consider new information.

The Elo rating system has been shown to provide good prediction accuracy (Nižnan et al., 2015), and better performance than standard PFA and BKT when extended in combination with PFA (Papousek et al., 2014). The Elo rating system provides both simplicity and flexibility in its student modeling approach which makes it easy to implement in educational systems (Pelánek, 2016; Von Davier et al., 2019; Yudelson et al., 2019).

5.4 Model Assessment

Models such as BKT, PFA, DKT, and Elo are often assessed and validated in multiple ways. The most common way that these types of models are assessed is by building the model on one group of students, applying it to a new, unseen group of students, and testing the model's performance at predicting future performance for these students. Typically, performance on the n^{th} opportunity to demonstrate a skill is predicted from the model's state at the $(n\text{-}1)^{th}$ opportunity to demonstrate that skill. Most commonly, student-level cross-validation is used so that every student's data can be used in both training and test set. By doing so, it becomes possible to determine whether a model is generalizable. The resultant predictions are evaluated using multiple metrics – there is an ongoing and continuing debate in the field as to which of several metrics is best, including AUC ROC, RMSE, log likelihood, and others (Pardos & Yudelson, 2013; Pelánek, 2015). There is also debate as to whether evaluation should be conducted at the grain-size of all data points together (i.e. differentiating prediction across students and skills) or individual within-student or within-skill (i.e. differentiating the model's ability to determine when a student learns or when a specific skill is known) (Pelánek, 2017b). Note that a model's ability to differentiate whether a student has learned – by being able to predict when a student will demonstrate successful or unsuccessful performance – is somewhat different than common psychometric notions of reliability of assessment in that the knowledge states of students are changing as they are being assessed. If the student is learning – and in an adaptive learning system we hope they are – we should not hope that our assessments of that student will agree over time.

Although the most common form of assessment of this type of knowledge model is immediate future performance, many papers have also looked at whether the resultant models could predict performance on future tests of the same knowledge, administered through other modalities, such as paper post-tests (see, for instance, Corbett & Anderson, 1995; Sao Pedro et al., 2013). This practice can help to establish that the model is capturing a more general indicator of student knowledge. Models are often also studied to determine whether all items belong within the skill, through examining whether better fit is achieved if an item is excluded (Corbett &

Anderson, 1995; Stamper & Koedinger, 2011). One other key differentiator between models is the degree to which they infer a latent construct. Whereas BKT and Elo infer a latent construct (i.e., student knowledge, ability), which can be used in several ways, DKT and PFA are limited to predicting performance on specific items. This limitation means that, despite DKT's better performance than BKT's original articulation, it cannot be straightforwardly used for many of the purposes that BKT has been used for, such as predicting performance outside the learning system and use in models of engagement and affect (e.g. Pardos et al., 2014).

5.5 Adapting to Student Attributes

Once learner attributes (in this paper, student knowledge) are modeled by an adaptive learning system, the pedagogical model is designed to select problems or customize instructional content for the learner. Adaptive systems take into account the information in the student models and behave differently in terms of pedagogical mechanisms for different students or groups of students, aimed at providing individualized support for better learning.

An example of adaptation that is more complex than simple mastery learning is seen in Assessment and Learning in Knowledge Spaces (ALEKS), (Canfield, 2001; Hu et al., 2012) a web-based assessment and learning system that attempts to identify when a student is struggling because the student does not know a prerequisite skill, so that it can switch to supporting the student in learning the prerequisite skill. As the student progresses in learning topics, ALEKS updates its student model and uses that information to make a principled selection of what skills it asks the student to work on.

Several aspects of the learning experience can be adapted, including the content's complexity or difficulty (Murray & Arroyo, 2002; VanLehn, 2006), scaffolding and support (Koedinger & Corbett, 2006; Razzaq et al., 2005), the visual interface (Ahn & Brusilovsky, 2009; Jeremic et al., 2009), and feedback prompts such as hints, prompts or communications from pedagogical agents (Biswas et al., 2005; Graesser et al., 2007; Razzaq et al., 2005). While many of these systems have hand-coded intervention logic, some experimental systems go beyond this to use algorithms which attempt to automatically discover which types of interventions work for which students, under which conditions (e.g. Chi et al., 2011). Through this process, the system "learns" more about the student in real-time, and what works for this student, as the student responds to the instructional or learning material presented to them.

Finally, some systems provide support through natural language dialogues. The best known example is AutoTutor (Graesser et al., 2007). It employs semantic analyses to compare a student's input text to the text of the ideal answer to a problem, as well as sample text representing other potential answers, including both

good answers and misconceptions. By recognizing a range of potential answers (and being able to do textual transformations between canonical answers and alternate ways of saying the same thing), AutoTutor can respond in a flexible and sensitive fashion. It adaptively responds to the student answers, whether answers are incomplete or representing misconceptions, by engaging in a variety of tutorial strategies, including asking for more information, giving positive, neutral or negative messages, prompts for missing words, giving hints, answering questions from students, and summarizing answers (Graesser et al., 2007).

5.6 Conclusion

In this chapter, we provided an overview of student modeling used in adaptive learning, focusing on modern approaches to inferring student knowledge – the most common student attribute being adapted to in current large-scale adaptive learning systems. We have discussed a small number of established adaptive educational systems, as well as established and more recent knowledge inference models. Although not discussed in detail here, most of the systems presented in this chapter have been shown to lead to significantly better student outcomes than traditional curricular approaches (Graesser et al., 2007; Hu et al., 2012; Koedinger et al., 1997; Leelawong & Biswas, 2008; Razzaq et al., 2005), although relatively few systems have achieved a high rating in research review databases such as the What Works Clearinghouse or Evidence for ESSA. Especially in the United States, adaptive learning systems have achieved scale, with systems such as the Cognitive Tutor or ALEKS being used by hundreds of thousands of students.

One interesting limitation to the current generation of adaptive learning systems, discussed in detail in (Baker, 2016), is that with a few exceptions (such as Affective AutoTutor, for instance), most existing systems adapt to a single dimension of the learner, in a single way. Individual systems become very effective at using mastery learning, or at providing feedback when students violate constraints, and perhaps giving messages as to why the student is wrong – but comprehensive support for problem selection, misconceptions, affect, motivation, disengaged behavior, and self-regulated learning is rare. Baker (2016) suggests that this is because of the difficulty and complexity of getting any one type of intervention correct; for adaptive learning systems to reach their full potential, they will have to surmount this challenge and consider and adapt to the student in a richer, more multi-dimensional fashion.

References

Ahn, J. W., & Brusilovsky, P. (2009). Adaptive visualization of search results: Bringing user models to visual analytics. *Information Visualization, 8*(3), 167–179.

Aleven, V., McLaren, B., Roll, I., & Koedinger, K. R. (2006). Toward meta-cognitive tutoring: A model of help seeking with a cognitive tutor. *International Journal of Artificial Intelligence in Education, 16*, 101–128.

Arroyo, I., Woolf, B. P., Burelson, W., Muldner, K., Rai, D., & Tai, M. (2014). A multimedia adaptive tutoring system for mathematics that addresses cognition, metacognition and affect. *International Journal of Artificial Intelligence in Education, 24*(4), 387–426.

Azevedo, R., Landis, R. S., Feyzi-Behnagh, R., Duffy, M., Trevors, G., Harley, J. M., Bouchet, F., Burlison, J., Taub, M., Pacampara, N., Yeasin, M., Rahman, A. K. M. M., Tanveer, M. I., & Hossain, G. (2012). The effectiveness of pedagogical agents' prompting and feedback in facilitating co-adapted learning with MetaTutor. In *International conference on intelligent tutoring systems* (pp. 212–221). Springer.

Baker, R., & Siemens, G. (2014). Educational data mining and learning analytics. In K. Sawyer (Ed.), *Cambridge handbook of the learning sciences* (2nd ed., pp. 253–274). Cambridge University Press.

Baker, R. S. (2016). Stupid tutoring systems, intelligent humans. *International Journal of Artificial Intelligence in Education, 26*(2), 600–614.

Baker, R. S. J. D., Corbett, A. T., & Aleven, V. (2008). More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing. In E. Aimeur & B. Woolf (Eds.), *Proceedings of the 9th International Conference on Intelligent Tutoring Systems* (pp. 406–415).

Beck, J., Chang, K. M., Mostow, J., & Corbett, A. (2008). Does help help? Introducing the Bayesian evaluation and assessment methodology. In *Intelligent tutoring systems* (pp. 383–394). Springer.

Biswas, G., Leelawong, K., Schwartz, D., Vye, N., & The Teachable Agents Group at Vanderbilt. (2005). Learning by teaching: A new agent paradigm for educational software. *Applied Artificial Intelligence, 19*(3–4), 363–392.

Brusilovsky, P., Karagiannidis, C., & Sampson, D. (2004). Layered evaluation of adaptive learning systems. *International Journal of Continuing Engineering Education and Life Long Learning, 14*(4–5), 402–421.

Canfield, W. (2001). ALEKS: A Web-based intelligent tutoring system. *Mathematics and Computer Education, 35*(2), 152.

Chi, M., Van Lehn, K., Litman, D., & Jordan, P. (2011). An evaluation of pedagogical tutorial tactics for a natural language tutoring system: A reinforcement learning approach. *International Journal of Artificial Intelligence in Education, 21*(1), 83–113.

Conati, C., Gertner, A. S., VanLehn, K., & Druzdzel, M. J. (1997, January). On-line student modeling for coached problem solving using Bayesian networks. In *User modeling* (pp. 231–242). Springer.

Corbett, A. T., & Anderson, J. R. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction, 4*(4), 253–278.

Craig, S. D., Anderson, C., Bargagloitti, A., Graesser, A. C., Okwumabua, T., Sterbinsky, A., & Hu, X. (2011, January). Learning with ALEKS: The impact of students' attendance in a mathematics after-school program. In *Artificial Intelligence in Education* (pp. 435–437). Springer.

D'Mello, S., Craig, S., Fike, K., & Graesser, A. (2009). Responding to learners' cognitive-affective states with supportive and shakeup dialogues. In *International Conference on Human-Computer Interaction* (pp. 595–604). Springer.

D'Mello, S., Lehman, B., Sullins, J., Daigle, R., Combs, R., Vogt, K., Perkins, L., & Graesser, A. (2010). A time for emoting: When affect-sensitivity is and isn't effective at promoting deep learning. In *Intelligent tutoring systems* (pp. 245–254). Springer.

DeFalco, J. A., Rowe, J. P., Paquette, L., Georgoulas-Sherry, V., Brawner, K., Mott, B. W., Baker, R. S., & Lester, J. C. (2018). Detecting and addressing frustration in a serious game for military training. *International Journal of Artificial Intelligence in Education, 28*(2), 152–193.

Embretson, S. E., & Reise, S. P. (2000). *Item response theory for psychologists*. Erlbaum.

González-Brenes, J., Huang, Y., & Brusilovsky, P. (2014). General features in knowledge tracing to model multiple subskills, temporal item response theory, and expert knowledge. In *The 7th International Conference on Educational Data Mining* (pp. 84–91).

Graesser, A. C., Jackson, G. T., & McDaniel, B. (2007). AutoTutor holds conversations with learners that are responsive to their cognitive and emotional states. *Educational Technology, 47*, 19–22.

Hu, X., Craig, S. D., Bargagliotti, A. E., Graesser, A. C., Okwumabua, T., Anderson, C., Cheney, K. R., & Sterbinsky, A. (2012). The effects of a traditional and technology-based after-school setting on 6th grade student's mathematics skills. *Journal of Computers in Mathematics and Science Teaching, 31*(1), 17–38.

Jeremic, Z., Jovanovic, J., & Gaševic, D. (2009). Evaluating an intelligent tutoring system for design patterns: The DEPTHS experience. *Educational Technology & Society, 12*(2), 111–130.

Karampiperis, P., & Sampson, D. (2005). Adaptive learning resources sequencing in educational hypermedia systems. *Educational Technology & Society, 8*(4), 128–147.

Khajah, M., Lindsey, R. V., & Mozer, M. C. (2016). How deep is knowledge tracing?. *arXiv preprint arXiv: 1604.02416*.

Khajah, M. M., Huang, Y., González-Brenes, J. P., Mozer, M. C., & Brusilovsky, P. (2014). Integrating knowledge tracing and item response theory: A tale of two frameworks. In *Proceedings of Workshop on Personalization Approaches in Learning Environments (PALE 2014) at the 22th International Conference on User Modeling, Adaptation, and Personalization* (pp. 7–12).

Koedinger, K. R., Anderson, J. R., Hadley, W. H., & Mark, M. A. (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education, 8*, 30–43.

Koedinger, K. R., & Corbett, A. T. (2006). Cognitive Tutors: Technology bringing learning science to the classroom. In K. Sawyer (Ed.), *The Cambridge Handbook of the Learning Sciences* (pp. 61–78).

Koedinger, K. R., Corbett, A. T., & Ritter, S. (2000). *Carnegie learning's cognitive tutor: Summary research results*. Carnegie Learning.

Leelawong, K., & Biswas, G. (2008). Designing learning by teaching agents: The Betty's brain system. *International Journal of Artificial Intelligence in Education, 18*(3), 181–208.

Mislevy, R. J., & Riconscente, M. M. (2006). Evidence-centered assessment design. In *Handbook of test development* (pp. 61–90). Erlbaum.

Mitrovic, A. (1998). A knowledge-based teaching system for SQL. In *Proceedings of ED-MEDIA* (Vol. 98, pp. 1027–1032).

Murray, T., Arroyo, I. (2002) Toward measuring and maintaining the zone of proximal development in adaptive instructional systems. *Proceedings of the International Conference on Intelligent Tutoring Systems,* 749–758.

Nižnan, J., Pelánek, R., & Rihák, J. (2015). Student models for prior knowledge estimation. *International Educational Data Mining Society*.

Nkambou, R., Mizoguchi, R., & Bourdeau, J. (Eds.). (2010). *Advances in intelligent tutoring systems* (Vol. 308). Springer Science & Business Media.

Ohlsson, S.: 1994, Constraint-based student modeling. *Student Modeling: the Key to Individualized Knowledge-based Instruction*, pp. 167–189.

Pane, J. F., Griffin, B. A., McCaffrey, D. F., & Karam, R. (2014). Effectiveness of cognitive tutor algebra I at scale. *Educational Evaluation and Policy Analysis, 36*(2), 127–144.

Papousek, J., Pelánek, R., & Stanislav, V. (2014, July). Adaptive practice of facts in domains with varied prior knowledge. In *Educational Data Mining 2014.*

Pardos, Z., & Heffernan, N. (2011). KT-IDEM: Introducing item difficulty to the knowledge tracing model. User Modeling, Adaption and Personalization, 243–254.

Pardos, Z. A., Baker, R. S., San Pedro, M., Gowda, S. M., & Gowda, S. M. (2014). Affective states and state tests: Investigating how affect and engagement during the school year predict end-of-year learning outcomes. *Journal of Learning Analytics, 1*(1), 107–128.

Pardos, Z. A., Baker, R. S. J. D., Gowda, S. M., & Heffernan, N. T. (2011). The sum is greater than the parts: Ensembling models of student knowledge in educational software. *SIGKDD Explorations, 13*(2), 37–44.

Pardos, Z. A., & Heffernan, N. T. (2010). Modeling individualization in a bayesian networks implementation of knowledge tracing. In *User modeling, adaptation, and personalization* (pp. 255–266). Springer.

Pardos, Z. A., & Yudelson, M. (2013, July). Towards Moment of Learning Accuracy. In *AIED Workshops.*

Pavlik Jr, P. I., Cen, H., & Koedinger, K. R. (2009). Performance factors analysis – A new alternative to knowledge tracing. *Online Submission*.

Pelánek, R. (2015). Metrics for evaluation of student models. *Journal of Educational Data Mining, 7*(2), 1–19.

Pelánek, R. (2016). Applications of the Elo rating system in adaptive educational systems. *Computers & Education, 98*, 169–179.

Pelánek, R. (2017a). Bayesian knowledge tracing, logistic models, and beyond: An overview of learner modeling techniques. *User Modeling and User-Adapted Interaction, 27*(3–5), 313–350.

Pelánek, R. (2017b). Measuring predictive performance of user models: The details matter. In *Adjunct Publication of the 25th conference on user modeling, adaptation and personalization (UMAP '17)* (pp. 197–201). ACM.

Piech, C., Bassen, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L. J., & Sohl-Dickstein, J. (2015). Deep knowledge tracing. In *Advances in neural information processing systems* (pp. 505–513).

Razzaq, L., Feng, M., Nuzzo-Jones, G., Heffernan, N. T., Koedinger, K. R., Junker, B., Ritter, S., Knight, A., Aniszczyk, C., Choksey, S., Livak, T., Mercado, E., Turner, T.E., Upalekar, R., Walonoski, J.A., Macasek, M.A., & Rasmussen, K.P. (2005). The Assistment project: Blending assessment and assisting. In *Proceedings of the 12th Annual Conference on Artificial Intelligence in Education* (pp. 555–562).

Rebolledo-Mendez, G., du Boulay, B., & Luckin, R. (2005, May). "Be bold and take a challenge": Could motivational strategies improve help-seeking. In Proceedings of the *2005 conference on Artificial Intelligence in Education: Supporting Learning through Intelligent and Socially Informed Technology* (pp. 459–466).

Reye, J. (2004). Student modelling based on belief networks. *International Journal of Artificial Intelligence in Education, 14*(1), 63–96.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1985). *Learning internal representations by error propagation* (No. ICS-8506). California Univ San Diego La Jolla Inst for Cognitive Science.

Rupp, A. A., Templin, J., & Henson, R. A. (2010). *Diagnostic measurement: Theory, methods, and applications*. Guilford Press.

Sao Pedro, M. A., Baker, R. S. J. D., Gobert, J., Montalvo, O., & Nakama, A. (2013). Leveraging machine-learned detectors of systematic inquiry behavior to estimate and predict transfer of inquiry skill. *User Modeling and User-Adapted Interaction, 23*(1), 1–39.

Scruggs, R., Baker, R.S., McLaren, B.M. (2020) Extending deep knowledge tracing: Inferring interpretable knowledge and predicting post system performance. *Proceedings of the 28th International Conference on Computers in Education*.

Sottilare, R., Holden, H., Graesser, A. C., & Hu, X. (2013). *Design recommendations for adaptive intelligent tutoring systems: Learner modeling* (Vol. 1). US Army Research Laboratory.

Stamper, J., & Koedinger, K. (2011). Human-machine student model discovery and improvement using DataShop. In *Artificial intelligence in education* (pp. 353–360). Springer.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction*. MIT Press.

VanLehn, K. (1996). Conceptual and meta learning during coached problem solving. In *Intelligent tutoring systems* (pp. 29–47). Springer.

VanLehn, K. (2006). The behavior of tutoring systems. *International Journal of Artificial Intelligence in Education, 16*(3), 227–265.

Von Davier, A. A. (2017). Computational psychometrics in support of collaborative educational assessments. *Journal of Educational Measurement, 54*(1), 3–11.

Von Davier, A. A., Deonovic, B. E., Yudelson, M., Polyak, S., & Woo, A. (2019). Computational psychometrics approach to holistic learning and assessment systems. *Frontiers in Education, 4*, 69.

Wenger, E. (1987). *Artificial intelligence and tutoring systems. Computational and cognitive approaches to the communication of knowledge*. Morgan Kaufmann.

Woolf, B. P. (2010). *Building intelligent interactive tutors: Student-centered strategies for revolutionizing e-learning*. Morgan Kaufmann.

Yeung, C.-K., & Yeung, D.-Y. (2018). Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*, 1–10.

Yudelson, M., Rosen, Y., Polyak, S., de la Torre (2019). Leveraging skill hierarchy for multi-level modeling with Elo rating system. In *Proceedings of the Sixth ACM Conference on Learning @ Scale*. IL: Chicago.

Yudelson, M. V., Koedinger, K. R., & Gordon, G. J. (2013, July). Individualized bayesian knowledge tracing models. In *International conference on artificial intelligence in education* (pp. 171–180). Springer.

Zhang, J., Shi, X., King, I., and Yeung, D.-Y. (2017). Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th International Conference on World Wide Web*, 765–774.

Part II
Methodology

Chapter 6
Concepts and Models from Psychometrics

Robert J. Mislevy and Maria Bolsinova

Abstract The concepts and methods of psychometrics originated under trait and behavioral psychology, with relatively simple data, used mainly for purposes of prediction and selection. Ideas emerged over time that nevertheless hold value for the new psychological perspectives, contexts of use, and forms of data and analytic tools we are now seeing. In this chapter we review some fundamental models and ideas from psychometrics that can be profitably reconceived, extended, and augmented in the new world of assessment. Methods we address include classical test theory, generalizability theory, item response theory, latent class models, cognitive diagnosis models, factor analysis, hierarchical models, and Bayesian networks. Key concepts are these: (1) The essential nature of psychometric models (observations, constructs, latent variables, and probability-based reasoning). (2) The interplay of design and discovery in assessment. (3) Understanding the measurement issues of validity, reliability, comparability, generalizability, and fairness as social values that pertain even as forms of data, analysis, context, and purpose evolve.

6.1 Introduction

Broadly speaking, psychometrics is the development and use of observational procedures and quantitative analysis to measure psychological attributes such as aptitudes, attitudes, and abilities. The goal of the present volume is to advance this quest in the face of vastly expanded opportunities for gathering data in rich digital environments, applying insights from contemporary psychology, harnessing recent developments in analytical and statistical methodology, and applying the results

R. J. Mislevy (✉)
Educational Testing Service, Princeton, NJ, USA
e-mail: rmislevy@umd.edu

M. Bolsinova
Tilburg University, Tilburg, The Netherlands
e-mail: m.a.bolsinova@tilburguniversity.edu

© Springer Nature Switzerland AG 2021

A. A. von Davier et al. (eds.), *Computational Psychometrics: New Methodologies for a New Generation of Digital Learning and Assessment*, Methodology of Educational Measurement and Assessment, https://doi.org/10.1007/978-3-030-74394-9_6

to pressing challenges in education, psychology, and other social and behavioral domains.

The preceding sentence actually describes the leading edge of psychometric research at every point in its century-and-a-half history, for each decade encounters fresh developments in psychology, technology, and analytical methods. Today's challenges and opportunities are unprecedented in scale and ambition, outstripping familiar methods on several fronts. But so it seemed 20 years ago, and 60 years ago, and 100 years ago. Our goal in this chapter is twofold. Our first aim is to describe some of the key psychometric models that have been developed, partly because they can prove useful in some more complex situations than those for which they were introduced, sometimes as originally used but sometimes with re-interpretation or extension. The armamentarium of psychometric methods up to this point is broader than is generally appreciated. But our second aim is to bring out deeper concepts that underlie more familiar methods, for they remain useful to offer insights and guide practice in hitherto unexplored territory.

To this end, in Sect. 6.2 we briefly describe three key concepts. Consistent with the rest of the volume, the emphasis is on educational assessment, but the concepts and models apply more broadly. The concepts we address are the essential nature of psychometric models (observations, constructs, narrative structures, latent variables, and probability-based reasoning), the interplay of design and discovery, and measurement issues as social values (specifically, validity, reliability, comparability, generalizability, and fairness; see Chap. 2).

In Sect. 6.3 we describe a number of psychometric models currently in use, present their basic equations, point out relationships to the key concepts, and note connections with the computational psychometric developments described in following chapters. The models addressed in this section are classical test theory (CTT); generalizability theory; item response theory (IRT), including its extension to multivariate and explanatory IRT: latent class modeling; cognitive diagnosis modeling; and factor analysis. Others, including hierarchical models and Bayesian networks, are mentioned in passing in this section and the one that follows it.

In Sect. 6.4 we note a more general development in statistics that is especially felicitous for computational psychometrics: assembling statistical models from components that reflect recurring relationships in a domain (Gelman et al. 2013). In assessment applications, models can be flexibly assembled from an open-ended collection of model fragments that encompasses the historical psychometric models mentioned above and others from domains including mathematical psychology, dynamic modeling, and complex systems analysis. These capabilities suit new forms of assessment such as simulation and game-based assessments (see Chap. 4 for background on such assessments, and Chaps. 3, 5, and 9–14 for examples of computational psychometric methods in these environments).

We close in Sect. 6.5 with thoughts on the value of concepts and models from the psychometric paradigm for use in assessment going forward, in a world of rapid advances in psychology, technology, and analytic methods.[1]

6.2 Key Concepts

In the following sections we discuss concepts that have been developed in psychometrics over the past century with simpler performances, psychological grounding, and forms of data. Although they have been instantiated in familiar psychometric models and practices, their importance extends directly to the more complicated settings addressed in this volume.

6.2.1 The Nature of Psychometric Models

Psychometric models use mathematical structures to model patterns in observable data that are considered to manifest peoples' psychological capabilities or tendencies in relevant situations. A descriptive model, such as cluster analysis or multidimensional scaling, seeks to discover patterns in the data; the resulting lower-dimensional descriptions are examined to uncover psychologically meaningful characteristics, or at least similarities, among persons, situations, or their interrelationships. A generative model, such as an IRT or latent class model, puts forward unobservable or latent variables associated with persons, upon which observable variables such as item responses are posited to depend on scholastically through a so-called link function. The kind of models and forms of link functions in a given application depend on the nature of the data, the targeted inferences, and the theory (or lack thereof) guiding the project.

A central concept in generative models is *conditional independence*. Commonly, observable variables are modeled as independent given latent variables (with theoretically-motivated exceptions that will be noted). Probability models and conditional independence permit a great flexibility in design—in building, selecting, adapting, and perceiving evidence in complex task situations—while synthesizing

[1]Most of the other methodological chapters provide data and computer code for examples. The R or Python codes for those chapters can be found at the GitHub repository of this book, https://github.com/jgbrainstorm/computational_psychometrics. This chapter is instead meant to survey a large number of models and discuss underlying concepts. Fortunately, the literature offers many examples, tutorials, and more technical presentations on psychometric models. Many useful R packages are freely available for the models we discuss. The CRAN project web site maintains a comprehensive listing and brief descriptions of such resources, at https://cran.r-project.org/web/views/Psychometrics.html.

the information they hold for the targeted aspects of students' capabilities (see especially Sects. 6.3.4 and 6.4.2).

6.2.2 Design and Discovery

Although a psychometrician is sometimes called into a project after data have already been collected, the preferred practice is for the modeling considerations to enter earlier, going back to issues such as theory, purpose, data collection, and interface design. This is especially the case in educational assessment, and all the more so as theories and data increase in size and complexity. A psychometric model plays a role in a larger argument (Chap. 2), connecting observable data with unobservable constructs such as knowledge, skills, and strategies. That the data can support the intended inferences reliably and validly depends on a clear understanding of the relationship among cognition, observation, and interpretation—the "assessment triangle" (National Research Council 2001).

The "interpretation" vertex in the assessment triangle contains the psychometric model, but its meaning, and the success of inference through it, depends on the quality of argumentation and the design decisions throughout the argument. The determination of interface, data, and model are thus often iterative, again all the more so with complex assessments, with results from earlier testing feeding back to improve the model, the data capture, the examinee interface, or the construct definition. Writing in the context of virtual performance based assessments, Mislevy et al. (2012) referred to this interplay as the dialectic between design and discovery.

6.2.3 Measurement Issues as Social Values

It was noted in Chap. 2 that the concepts of validity, reliability, fairness, comparability, and generalizability are "not just measurement issues, but *social values* that have meaning and force outside of measurement wherever evaluative judgments and decisions are made" (Messick 1994, p. 13; emphasis original). Psychometric models are instrumental in investigating these properties of an assessment; reliability in the form of evidence about an examinee in terms of estimation error, for instance; fairness as equivalent operating characteristics across demographic groups; and validity in the form of empirical evidence for the quality of decisions made through the model. Procedures developed to characterize these properties in familiar assessments have been developed over the years for the kinds of assessments and psychometric models discussed in the following section. They provide metrics to examine these critical aspects of the quality of an assessment in use, and to guide further improvements. Extending the ideas into new forms of assessment and psychometric methods is fundamental to computational psychometrics.

6.3 Psychometric Models

6.3.1 Common Notation

When presenting different psychometric models we will use common notation for the parts of the model shared by all psychometric models.[2] A psychometric model is typically describing the relationship between the observed variables (often a test score or an item score) and the unobserved (latent) variable that one intends to measure. We denote the possibly vector-valued latent variable by θ and the observed variables by X; index i is used to identify the tasks in an assessment, such as items in a test.

6.3.2 Classical Test Theory

Classical test theory (CTT), pioneered by Charles Spearman (1904) and formalised in Lord and Novick (1968), was the first theory of measurement of psychological constructs using tests. A focus of CTT is the reliability of the test scores. Reliability is the quality of test which refers to the consistency of test results over replications. Reliability can be derived from the fundamental equation of CTT:

$$X = \theta + \epsilon; \tag{6.1}$$

that is, the observed test score (X) is equal to the sum of the true score (θ, the latent variable of interest) and measurement error (ϵ). The measurement error is assumed to have a mean of zero and to be unrelated to the true score. Therefore, the true score can be defined as the expected value of the observed test score over replications. The total variance of the observed score is decomposed into the variance of the true score and the error variance:

$$\sigma_X^2 = \sigma_\theta^2 + \sigma_\epsilon^2. \tag{6.2}$$

Reliability of the test score is defined as the proportion of the true variance in the total variance:

$$\rho = \frac{\sigma_\theta^2}{\sigma_\theta^2 + \sigma_\epsilon^2}. \tag{6.3}$$

[2]Note that for some of the models, this notation is not the one typically used within that modeling framework.

The smaller the measurement error is, the higher the reliability of the test is. Different ways of estimating reliability have been developed within CTT including computing the correlation of the test scores from multiple administrations of a test in a collection of examinees, the correlation of parallel forms of the same instrument, and, when a test consists of independent items, various measures of internal consistency. The standard error of measurement can be approximated using the relationship

$$\sigma_\epsilon^2 = (1 - \rho)\sigma_X^2. \tag{6.4}$$

From the simple premises of CTT follow a great many practical methods for working with familiar test forms—addressing not only reliability itself, but, among others, predictive validity, optimal use of multiple measures, and adjustments for the effects of measurement error on correlations among variables (Gulliksen 2013). For an ongoing testing program based on the production of parallel test forms, the latent true-score scale provides a common metric for reporting test scores for examinees who have been administered different forms—a "psychometric backbone" for characterizing examinees across different observational conditions. Not *that* different, of course; parallel test forms must satisfy strong constraints as to test length, item content, and administration conditions. Nevertheless, CTT was the first appearance of using latent variable models as a mechanism to synthesize evidence about examinees in an interpretative frame that accommodates disparate bodies of evidence. Other psychometric models discussed below extend the idea further.

The original expression of the CTT link function, Eq. (6.1), does not specify its functional form, which would be written as $p(X|\theta)$; a great many useful results have been derived using only means, variances, and correlations (Gulliksen 2013). Full specification of the link function, say as $X \sim \mathcal{N}(\theta, \sigma_\epsilon^2)$, enables likelihood and Bayesian inference.

Many CTT methods have counterparts in more complex forms of assessment, and CTT can offer insights into their development. In particular, CTT introduces two indispensable reliability concepts in a most basic form. First, whenever any attribute of a person or performance could be measured more than once, no single measure should be accepted uncritically as definitive. One learns about the quality of evidence that a measure provides only by seeing how interchangeable measures of the same kind vary from one another. Taking their hypothetical average over many such observations as the target of interest—a latent variable—provides a means to take this uncertainty into account. Complex assessments such as games can face this question in multiple locations, each of which affects the quality of inferences (Chap. 4). Second, the concept of a standard error of measurement can be extended more generally to model-based estimates of latent variables in more complicated models, such as those described below and in successive chapters, to characterize the weight of evidence for inferences about examinees' capabilities, as expressed through the model.

6.3.3 Generalizability Theory

Generalizability theory (Cronbach et al. 1972) can be seen as an extension of CTT, as it also concerns the reliability of the measures, but additionally aims at characterizing different sources of measurement error rather than assuming a single undifferentiated source. Generalizability theory acknowledges that measurement of a particular construct may be executed in a variety of conditions, for example with different instruments, in different conditions, at different time points, and with different raters. The range of possible conditions under which the construct can be measured is referred to as the universe of admissible observations, and each observation is assumed to be a sample from this universe. The goal of generalizability studies is to quantify the measurement error and its different sources. A linear model is typically used to model the effects of different aspects of measurement, called facets (e.g., tasks, occasions, raters) and their contributions to measurement error. Here is a simple example for multiple tasks crossed with multiple time points:

$$X_{jk} = \theta + \alpha_i + \beta_j + \delta_{ij} + \epsilon_{ij}, \tag{6.5}$$

where X_{jk} is the observed score obtained using instrument j on measurement occasion k, θ is the universe score, α_i and β_j are the effects of the i-th instrument and the j-th occasion, respectively, δ_{ij} is the interaction effect and ϵ_{ij} is the error. The variance of the observed score across the universe of admissible observations is decomposed into a sum of the facet components:

$$\sigma_X^2 = \sigma_\theta^2 + \sigma_\alpha^2 + \sigma_\beta^2 + \sigma_\epsilon^2, \tag{6.6}$$

where the variance of the interaction term cannot be separated from the residual variance and the two are combined in the residual variance σ_ϵ^2.

The idea of modeling components of uncertainty that correspond to facets extends in terms of both numbers of facets such as multiple raters and multiple correlated dimensions of rating of complex performance, and to conditions of observational designs, such as different possible combinations of crossing and nesting when multiple raters are available for scoring multiple dimensions of multiple tasks. Each combination of scoring design, such as crossing or nesting raters within or across tasks or rating dimensions, has different implications for both operational costs and the accuracy of inferences.

Generalizability theory provides a framework for calculating what corresponds to standard errors of measurement, but as associated with different inferences that can be drawn from the same observations, such as comparing one examinee against a fixed criterion, or one comparing one examinee to another if they took the same tasks, or if they took different tasks, or if they were scored by, say, the two same raters or by three different randomly selected raters. The framing has shifted from "the reliability of a measure" to the more general, and more extensible, framing of

"evidentiary value of data obtained in a particular way for targeted inferences." The concern has been generalized from "estimating the standard error of measurement" to "quantifying the evidence for given claims."

As its very name suggests, generalizability theory has been used to study the measurement issue of generalizability; that is, how the information in given task performances generalizes to inferences about students' capabilities for tasks that differ in various ways. This issue is particularly important for assessments involving higher-level skills and complex performance tasks, which are set in a given context and draw on particular knowledge and representations. How much does a student's performance in a collaborative hands-on problem-solving task on sow-bug behavior tell us about her prospects for a problem about paper towel absorbency, with different collaborators, in a computer simulation? The answer is usually "less than you think" (Gao et al. 1994). A rich authentic performance task may provide copious data and sophisticated analysis may distill keen insights—about performance in this task, in this context, on this topic, on this occasion. The extent to which it provides evidence more broadly is always an empirical question. Generalizability theory was the first systematic machinery for exploring these issues in the behavioral sciences.

Generalizability theory provides these extensions of measurement concepts in the specific forms of linear models and a presumed-constant random error term (i.e., a CTT link function). The same evidentiary advances have subsequently been incorporated into the broader frameworks of linear and nonlinear models for relationships among observations and facets, and to link functions that are suited to data in the form of counts, response times, ordered scales, and categorical observations, as well as combinations of forms of data (Rabe-Hesketh & Skrondal 2004). A frontier of computational psychometrics lies in incorporating these ideas to increasingly complex observational settings.

6.3.4 Item Response Theory

Development of IRT as a model-based alternative to CTT started in 1950s, pioneered by Frederic Lord, Georg Rasch, and Paul Lazarsfeld. In IRT the distribution of the observed item scores is modeled conditional on a latent variable that is assumed to be continuous. Under the Rasch model (Rasch 1960), one of the most commonly used IRT models, the probability of a correct response to an item is given as

$$\Pr(X_i = 1 \mid \theta) = \frac{\exp(\theta - \beta_i)}{1 + \exp(\theta - \beta_i)}, \tag{6.7}$$

where θ is a person ability parameter and β_i is the difficulty of the item. The latter can be interpreted as the value on the latent variable scale for which the probability of a correct response is equal to the probability of an incorrect response. The model can be extended by allowing the items to vary in their discrimination, and to take into account guessing on multiple-choice items, yielding the three-parameter logistic

IRT model (Birnbaum 1968):

$$\Pr(X_i = 1 \mid \theta) = \gamma_i + (1 - \gamma_i) \frac{\exp(\alpha_i(\theta - \beta_i))}{1 + \exp[\alpha_i(\theta - \beta_i)]}, \tag{6.8}$$

where γ_i and α_i are the guessing and the discrimination parameter of item i.

An IRT link function, such as (6.7) for the Rasch model, gives the conditional probability of the response to a particular item given θ. The assumption of conditional independence comes into play for the responses to multiple items:

$$p(x_1, \ldots, x_n \mid \theta) = \prod_i p(x_i \mid \theta). \tag{6.9}$$

It follows that associations among observational variables are thus determined strictly through θ, and evidence about θ can be obtained using different sets of items. Implications follow for both gauging reliability and implementing adaptive testing schemes.

The concepts of reliability and measurement precision are also important in IRT, but they are implemented differently than in CTT. Moreover, while in CTT it is typically assumed that measurement error is the same for everyone in the population, in IRT measurement error depends on the person's ability level and on which items have been administered to the person. Important concepts within IRT related to measurement error are item and test information (denoted by $I_i(\theta)$ and $I(\theta)$, respectively), which for example in the Rasch model are defined as follows:

$$I(\theta) = \sum_i I_i(\theta) = \sum_i \frac{\exp(\theta - \beta_i)}{(1 + \exp(\theta - \beta_i))^2}. \tag{6.10}$$

Item information is equal to the product of the probability of a correct response and the probability of an incorrect response, and is therefore the highest when both probabilities are equal to 0.5, which is the case for the value of ability equal to the item difficulty. Measurement error is also a function of the ability level. It is the inverse of the square root of the test information, or $I(\theta)^{-1/2}$.

The idea that test information for a particular person can be maximized by selecting items with difficulty levels close to the ability of the person is the basis for computerized adaptive testing (CAT). During CAT each next item for the respondent is selected on the basis of her responses to previous items in order to maximize item information (while also satisfying constraints that concern item content, format, item exposure, etc.). When an item has been selected to administer and a response is observed, its link-function is combined with previous belief about θ via Bayes theorem to produce a posterior distribution. After having already observed n items, for example, the resulting posterior distribution for θ after a response to the $(n+1)^{st}$ item is obtained as

$$p(\theta \mid x_1, \ldots, x_n, x_{n+1}) \propto p(\theta \mid x_1, \ldots, x_n)\, p(x_{n+1} \mid \theta). \tag{6.11}$$

This posterior distribution can then be used to further update belief with information from additional item responses or other sources, to select optimal items to administer, or to carry out inferences or decisions based on current knowledge about θ while taking the uncertainty into account.

CAT is a simple example of the modular assembly for probability-based inference in observational situations that psychometric latent-variable models afford. Figure 6.1 depicts the idea as it applies in IRT-CAT. At the left is a computational module that for θ maintains a probability distribution for a person's θ conditional on all information, if any, up to a given point, denoted $p(\theta \mid *)$. At the right is a collection of modules for each item in an item bank contain an IRT link function for that item, i.e., the conditional probability distribution for a response to that item given θ. Note that the θ in the item modules is dotted to indicate that the module does not contain a distribution for θ, only for the item-response conditional distribution depending on θ. Its dotted presence indicates that the module can however be "docked" with the θ module for calculations such as (6.11). Bayesian inference during an ongoing CAT can thus be carried out with local operations among such modules assembled to reflect the evolving state of belief as the test proceeds. Section 6.4.2 extends these ideas to more complex models and forms of assessment.

CAT is one of the examples of the flexibility of IRT in terms of test design. IRT is generally very flexible in terms of handling incomplete testing designs and the missing data that arise from different branching schemes (Mislevy & Wu 1996). However, different approaches to dealing with missing data in IRT and elsewhere

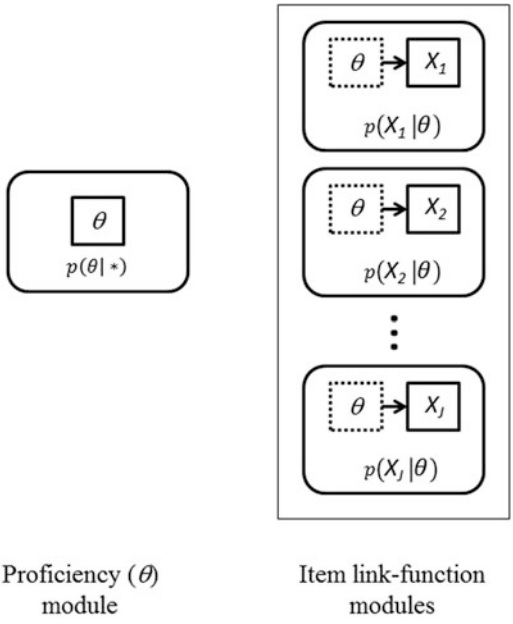


Fig. 6.1 Probability modules for IRT-based computerized adaptive testing

might be appropriate with different types of missingness, so-called nonignorable missing, as occur when examinees run out of time or when they do not answer because they believe they would probably answer incorrectly (Little & Rubin 2014; Mislevy 2016).

Since in IRT the focus of analysis is on the item level rather than on the level of the test score as in CTT, in IRT different approaches to important measurement concepts have been developed. One of the important contributions of IRT and latent variable modeling in general is the definition and formalisation of the concept of test fairness. When looking only on the test score as a whole it is impossible to determine whether the differences in the test scores are due to the true differences between the groups or due to the fact that the test puts one of the groups at a disadvantage. However, at the level of individual items, an important sense of test fairness can be defined more formally and tested with empirical data. When evaluating test fairness within IRT researchers investigate whether measurement invariance holds for all test items, where measurement invariance is defined as conditional independence of item scores of any background variables $\mathbf{Z}$ (e.g., gender, race, socioeconomic status) given the measured ability (e.g., Mellenbergh 1989; Meredith 1993; Millsap 2012):

$$\Pr(X_i = 1 \mid \theta, \mathbf{Z}) = \Pr(X_i = 1 \mid \theta), \tag{6.12}$$

That is, the probability of a correct response to the item conditional of ability is the same regardless group membership or any other characteristics of the respondent. The concept of measurement invariance has been also labeled as "measurement equivalence" (e.g., Byrne et al. 1989), "lack of item bias" (e.g., Mellenbergh 1989), and "absence of differential item functioning (DIF)" (e.g., Swaminathan & Rogers 1990). This formal statistical definition made it possible to develop a variety of methods for evaluating test fairness of a given test (e.g., Lord 1980; Thissen et al. 1988; Raju 1988; Swaminathan & Rogers 1990; Magis et al. 2010).

While the earliest IRT models were based on rather strict assumptions of unidimensionality, conditional independence, and stability of the latent variable throughout the testing situations, extensions have been developed that allow for multiple latent variables (Reckase 2009), residual dependencies between the items (e.g., in the testlet model of Bradlow et al. 1999), tree-like structures within response process (De Boeck & Partchev 2012), and dynamic change of ability (Embretson 1991; Wang et al. 2013). Some or all of these features can be found in the more complex forms of assessment such as those embedded in simulations and intelligent tutoring systems.

Many of the extensions of IRT models to more complex contexts became accessible and widely used once computationally efficient Bayesian estimation methods such as Markov chain Monte Carlo (MCMC) estimation have become available (Fox 2010). The same computational techniques have been gainfully applied with the other models discussed in this section and the current developments sketched in the section that follows. Section 6.4.2 offers more general remarks about this topic.

Performance on educational tests often depends on more than one ability or skill, which makes multidimensional IRT models relevant for measurement practice. Different types of multidimensional models have been developed within IRT, with one of the most important distinctions being between the compensatory and non-compensatory models (Way et al. 1988; Bolt & Lall 2003). In compensatory models the effect of the multiple abilities on the item scores are additive:

$$\Pr(X_i = 1 \mid \boldsymbol{\theta}) = \frac{\exp\left(\sum_k \alpha_{ik}\theta_k + \beta_i\right)}{1 + \exp\left(\sum_k \alpha_{ik}\theta_k + \beta_i\right)} \tag{6.13}$$

where $\boldsymbol{\theta}$ is a vector of abilities involved in solving the items on a test, α_{ik} determines the strength of the relationship between the k-th ability and the score on item i, and β_i is the intercept of item i which determines how often the item is responded to correctly. The model is compensatory in the sense that high values on one ability can compensate for low values on another ability. The idea behind non-compensatory models is that some items might consist of multiple steps which each require a separate ability, such that a low value of one of ability would mean that the probability of succeeding with the corresponding step of the solution and the probability of a correct response in general would be low even if the abilities involved in the other steps of the solution process are high. The probability of a correct response to an item related to two abilities in a non-compensatory model is the following:

$$\Pr(X_i = 1 \mid \theta_1, \theta_2) = \frac{\exp(\alpha_{i1}\theta_1 + \beta_{i1})}{1 + \exp(\alpha_{i1}\theta_1 + \beta_{i1})} \frac{\exp(\alpha_{i2}\theta_2 + \beta_{i2})}{1 + \exp(\alpha_{i2}\theta_2 + \beta_{i2})}; \tag{6.14}$$

that is, the final probability of a correct response is the product of the probabilities of succeeding on the two steps in the solution process, each of which is modeled by a two-parameter unidimensional IRT model.

A propitious development in IRT for more complex forms of assessment (along with the cognitive diagnosis models discussed in Sect. 6.3.5) is the formal incorporation of cognitive theory into task construction and psychometric modeling. An early example is the linear logistic test model (LLTM; Fischer 1973), an extension of the Rasch model (6.7) with linear constraints on the item difficulties:

$$\Pr(X_i = 1 \mid \theta) = \frac{\exp(\theta - \sum_j q_{ij}\eta_j)}{1 + \exp(\theta - \sum_j q_{ij}\eta_j)}, \tag{6.15}$$

where q_{ij} is the weight of the j-th item feature for item i, and η_j is the contribution to difficulty from the j-th feature. The encoding of items reflects features that are posited to evoke psychologically, instructionally, or socially relevant aspects of persons' capabilities. (A similar Q-matrix also appears in the cognitive diagnosis models discussed in Sect. 6.3.5.) In the LLTM, these item features drive item difficulty with respect to the unidimensional θ. In multivariate extensions, different

features can be modeled as drawing on different aspects of knowledge and ability in theory-driven ways (De Boeck & Wilson 2004). Whereas this reasoning may have previously been implicit in individual test developers' minds, it is now explicit, connected to cognitive theory, and integrated into the evidentiary-reasoning machinery of the psychometric model. This joint framing and modeling strengthens the validity argument for assessments that consist of items so constructed. Even with familiar assessments, then, IRT modeling can thus connect directly with the sociocognitive psychological perspective outlined in Chap. 2. We say more about extending this work to more advanced forms of assessment in the next section.

6.3.5 *Latent Class Analysis and Cognitive Diagnosis Models*

Introduced roughly at the same time as the first IRT models, another strand of latent variable modeling shares some important features with IRT, but differs in a key respect. In latent class models (Lazarsfeld 1959), the latent variables are taken to be discrete, to indicate qualitative differences between the respondents as opposed to the quantitative differences modeled in IRT. Within each latent class the observed scores on different items are assumed to be independent:

$$\Pr(X_i = 1, X_j = 1 \mid \theta = c) = \pi_{ic}\pi_{jc}, \tag{6.16}$$

where π_{ic} and π_{jc} are the probabilities of a correct response to items i and j, respectively, for persons in Class c. Based on the observed responses of a sample of respondents to a set of items, one can determine the number of latent classes (e.g., by using information criteria to compare models with different numbers of classes), the class sizes, and the item probability profiles within each class. Such analyses constitute exploratory latent class analyses; that is, analyses in which the number and characteristics of the latent classes are not specified a priori. Latent class analysis can also be performed in a confirmatory way, such that the number of classes and the patterns in the item probability profiles are specified based on theory, as for example was done by Jansen and van der Maas (2002) for Piaget's balance scale tasks.

The idea of distinct groups of respondents that differ in their expected response profiles has been further developed within educational measurement in the so-called cognitive diagnosis models (CDM; von Davier 2008; Henson et al. 2009; Rupp et al. 2010; De la Torre 2011). In these models the latent classes can be interpreted as groups of respondents that have or have not mastered certain skills needed to solve the items in the test. For example, it might be that for solving mathematics items in a particular subdomain three different skills are needed. These three skills define eight latent classes differing in the profile of mastery vs. non-mastery of these skills (from {000} for a non-master on all three skills to {111} for a master on all three skills).

The probabilities of a correct response to particular items depend on the skills profile of the person and on the so-called Q-matrix (Tatsuoka 1983). In the Q-matrix for each combination of an item with a skill the value of 1 indicates that this skill is required for solving the item, while the value of 0 indicates that it is not. The Q-matrix is usually specified a priori based on educational theory and test design, but methods for estimating the Q-matrix from observed data have also been developed recently (Chen et al. 2018; Liu et al. 2012). By way of example, one of the most commonly used CDMs is the DINA model (Junker & Sijtsma 2001) which specifies the probability of a correct response to an item in the following way:

$$\Pr(X_i = 1 \mid \boldsymbol{\theta}) = (1 - s_i) \prod_k \theta_k^{q_{ik}} + g_i \left(1 - \prod_k \theta_k^{q_i k}\right), \tag{6.17}$$

where $\boldsymbol{\theta}$ is the binary skill vector, q_{ik} indicates whether the k-th skill is required for solving item i, s_i is the slipping probability (i.e., the probability of giving an incorrect response even if all skills required for solving the item have been mastered) and g_i is the guessing probability (i.e., the probability of giving an incorrect response even if one or more skills required for solving the items have not been mastered).

6.3.6 *Bayesian Inference Networks*

The latent class models and cognitive diagnosis models of Sect. 6.3.5 can be expressed as Bayesian inference networks, or Bayes nets for short (Almond et al. 2015). A Bayes net is a joint probability distribution over a collection of discrete variables, say $(A_1, \ldots, A_n)$, and accompanying representations as an acyclic directed graph and a recursive representation of conditional probability distributions:

$$\begin{aligned}
&\Pr(A_n, A_{n-1}, \ldots, A_2, A_1) \\
&= \Pr(A_n | A_{n-1}, \ldots, A_2, A_1) \times \Pr(A_{n-1} | A_{n-2}, \ldots, A_2, A_1) \times \cdots \times \\
&\quad \Pr(A_2 | A_1) \times \Pr(A_1) \\
&= \prod_k \Pr(A_k | A_{k-1}, \ldots, A_1),
\end{aligned}$$

The graphical representation is convenient for building models with subject-matter experts and task developers. The recursive representation is of interest when many of the terms on the right sides of the conditioning bars drop out due to conditional independence relationships. In the graphical representation, which was fully connected to begin with, edges corresponding to dependencies drop out and the remaining graph can become much simpler (see Sect. 6.4.3). In favorable cases, calculations for updating belief when new information arrives become considerably faster than rote application of Bayes theorem. This is often the case in psychometric models, as observable variables X within given tasks are often presumed to be

conditionally independent given a relatively small collection of r latent variables θ. The conditional probabilities and θ distribution can be further modeled in terms of higher-level structures and parameters.

Bayes nets are widely used in applications such as troubleshooting, medical diagnosis, and intelligence analysis, for assembling and rapidly updating belief as new information arrives and decisions must be made about next steps (Sect. 6.4.3). Learning systems and assessment with interactive tasks as in games and simulations are such cases (Chaps. 3–5, 11–13).

The Bayes nets formulation also accommodates models that address change in latent variables θ over time. These are instances of dynamic Bayes nets (Murphy 2002), in that the values of one or more unobserved variables change across time points. There are standard Bayes nets for each successive time point; they are linked by variables that are pertinent at the time points at issue, some or all of which may have different perhaps-unknown values at different time points. A dynamic Bayes net for assessment is appropriate when students may be learning as they proceed through a task or a learning module. In such models, θ may change but is never observed, and observed responses X_t at each time point t depend only on the unknown value of θ_t. Such models are discussed in Chaps. 3–5, 12, and 13.

6.3.7 Factor Analysis

Another major psychometric approach is factor analysis. Its development started in the early twentieth century and continued throughout the century parallel to similar developments in IRT. As with CTT, the initial development of factor analysis can be attributed to Charles Spearman, who was studying the structure of human intelligence (Spearman 1904). The fundamental idea behind factor analysis is that the correlation between the observed variables (e.g., test scores on different cognitive tests) can be explained by individual differences in the underlying, possibly vector-valued, latent variables, called factors (e.g., in Spearman's studies, general intelligence; in reading research, subskills involved in comprehension).

Since factor models initially were developed for continuous observed variables, we will focus on continuous variables. It should be noted, however, that factor analytic models have been adapted to modeling categorical variables as well (Bartholomew 1980) and that some IRT models have been shown to be equivalent to factor models for categorical data (Takane & De Leeuw 1987). In a factor model the expected value of the observed variable X_i is a linear function of the factors:

$$\mathcal{E}(X_i \mid \boldsymbol{\theta}) = \nu_i + \boldsymbol{\lambda}_i^T \boldsymbol{\theta}, \tag{6.18}$$

where ν_i is the item intercept and $\boldsymbol{\lambda}_i$ is a vector of factor loadings specifying the relationships between the factors and X_i. The model-based covariance matrix of the observed variable, denoted by $\boldsymbol{\Sigma}$, is equal to:

$$\boldsymbol{\Sigma} = \boldsymbol{\Lambda}\boldsymbol{\Phi}\boldsymbol{\Lambda}^T + \boldsymbol{\Psi}, \tag{6.19}$$

where $\boldsymbol{\Lambda}$ is a matrix of factor loadings, $\boldsymbol{\Phi}$ is a matrix of factor covariances, and $\boldsymbol{\Psi}$ is a diagonal matrix of residual variances.

The factor loadings matrix indicates the presence and the strength of the relationship between the factors and the observed variables. In exploratory factor analysis the full factor loadings matrix is estimated freely (with the exception of some identification constraints), while in confirmatory factor analysis the structure of $\boldsymbol{\lambda}$ is pre-determined, typically by fixing some of the factor loadings to zero (analogous to the Q-matrix in CDMs). The goal of exploratory factor analysis is to investigate the structure of the relationship between the observed variables and explain it as much as possible with a few factors and a factor loadings matrix with a simple structure (e.g., most items having strong loadings only on one factor; Tucker 1955). Due to the possibility of factor rotations, however, the same $\boldsymbol{\Sigma}$ can be a result of multiple factor solutions. Confirmatory factor analysis is often used in the validation of measurement instruments to establish that the dimensionality and the structure of the measure is as intended by the test designers.

6.3.8 Network Analysis

Network analysis takes a different starting point than the individual-based, latent-variable models discussed above (although Sect. 6.4.6 discusses a computational-psychometric approach that synthesizes the two approaches). Applications of network analysis in psychometrics and sociology concern relations among units of some sort, such as persons, businesses, concepts, events, or behaviors (Wasserman & Faust 1994); i.e., social network analysis, or SNA. Mathematical concepts from graph theory prove useful to express patterns among social relationships, such as dominance, and properties of network relationships, such as transitivity. The basic form of a network is a collection of nodes X_i and edges that connect nodes, characterized by their weight, sign, and direction. Extensions include networks with time dimensions and multiple groups.

Practical challenges arise from the fact that theoretical properties are rarely exact in real-world circumstances, and the data available to study networks are noisy. Psychometricians have contributed significantly to network analysis by developing probability frameworks for estimating properties and testing hypotheses expressed in terms of networks. A more recent challenge is the immense increase in data and complexity of networks of interest, as occur among persons' actions and interactions in internet environments such as social networks and massively multiplayer online games (Chap. 13).

6.4 Toward Computational Psychometrics

As described in the preceding sections, many concepts and models have been developed in psychometrics that have proven their value for designing, analyzing, and supporting inferences from educational assessments. As discussed in Chap. 2, however, developments in psychology and digital technologies impel us toward assessments that are richer, more interactive, and better tuned to students' learning pathways. Challenges to familiar psychometric practices include conditional dependence, multimodal data, path dependence of students' actions, vast amounts of data about the details of students' actions, interactions among multiple students, increased diversity of student backgrounds, and higher-level targets of inference such as collaborative skills and scientific investigation.

In this section, we note six more recent developments that bridge traditional applications of psychometrics in educational assessment with new forms of assessment and analytic methods from other fields. These developments are stochastic Bayesian estimation methods, modular assembly of models, dynamic model construction, interfaces of psychometric and data analytic techniques, the interplay of exploratory and confirmatory analyses, and network psychometrics.

6.4.1 Stochastic Bayesian Estimation Methods

The history of estimation for the psychometric models described above exhibited distinct traditions within families of models and specialized approaches that capitalized on the structures and uses of the various models. Most were carried out under the frequentist inferential paradigm, using method of moments, least squares, or maximum likelihood estimators, and computational difficulties limited the complexity and flexibility with which models could be constructed and estimated. The foundations of the modern stochastic estimation methods, such as MCMC mentioned in Sect. 6.3.4 and discussed in greater detail in Chap. 7, are usually credited to Stanislaw Ulam and John von Neumann in the late 1940s, but it was not until the 1990s that the methods emerged widely and generally. They have revolutionized not simply how statisticians estimate parameters in given models, but how they think about modeling, uncertainty, and inference. So it is in psychometrics (Levy 2009).

Gelman et al. (2013) provide a comprehensive overview of stochastic Bayesian inference, but four key ideas can be recognized. First is the notion of assembling statistical models from components that reflect recurring relationships in a domain. Analysts assemble, from an open-ended collection of model fragments, probability-based reasoning machinery that captures recurring substantive and evidentiary patterns. Second, with such structures, it is often possible to express interdependence among variables in terms of less densely connected clusters of more densely connected variables. Significant conditional dependencies can result. Third,

a Bayesian framework enables an analyst to express what is known a priori about the variables from previous research or substantive theory, but more importantly it places the inferential problem into a proper joint distribution over all variables involved, both observable and latent. Fourth, inference about probability structures over such networks can be carried out with relatively simpler local calculations, typically moving stochastically through the network with one or a few variables at a time (the "Monte Carlo" reference) to consistently converge to, then continue to move around in and thus approximate, the full joint posterior distribution (a Markov chain) conditional on the observed data.

As noted in Sect. 6.3.4 and Chap. 7, MCMC methods have enabled many extensions of IRT models (Patz & Junker 1999). MCMC and more recently Hamiltonian Monte Carlo methods (Luo & Jiao 2018) make it possible to carry out inference with psychometric models with large numbers of dimensions and large numbers of parameters. Such methods provide for consistent and better conditioned inference in generalizability theory, cognitive diagnosis modeling, factor analysis, structural equations, and hierarchical models (Levy & Mislevy 2016). As discussed in the following section, they free researchers from the bounds of specific and constrained families of models and distributions—to instead construct models from components that better reflect the structures of the substantive domains of their problems. They provide a principled platform for inference even as data, models, and substantive knowledge continue to evolve, as seen in many of the following chapters.

6.4.2 Modular Assembly of Analytic Models

A recent development in statistical analysis that was mentioned in the preceding section is the notion of assembling statistical models from components that reflect recurring relationships in a domain (Gelman et al. 2013). Early applications of modular analysis in psychometrics were IRT-CAT, discussed above, and structural equation modeling (SEM). SEM allows users to custom-design their models yet carry out estimation in a common framework, drawing on structures from CTT, factor analysis, path analysis, and systems of linear equations (Joreskog et al. 1979). Subsequent developments include such effects as link functions for multiple kinds of input data, hierarchical structures as to both persons and observational settings, and incorporation of collateral variables for persons and situations, drawn from social and cognitive theories. It may be noted that most of the models presented in Sect. 6.3 can be seen as special cases of generalized linear latent variable models (Rabe-Hesketh & Skrondal 2004). The different types and structures for latent variables and the different types of observable variables and their relationships to the latent variables can be employed, as motivated by the application.

The model fragments that constitute psychometric models support reasoning about the inferential issues that arise from recurring features of observable situations, inferring cognitive capabilities from situated actions, and accounting for

hierarchical organization in social and psychological phenomena (Chap. 2). Seeing these fragments in their basic forms in familiar psychometric models adds intuition for contemporary developments.

6.4.3 Dynamic Model Construction

A relatively recent application of modular assembly of psychometric models has been in interactive digital assessments. The idea is to assemble probability-model fragments in real time for reasoning from observations x to latent variables θ, selecting or constructing fragments in light of the emerging features of the performance situation as examinees proceed. CAT with IRT is a direct application of these ideas. We point to applications with more advanced forms of assessment and a more complex psychometric model, namely the Bayes nets extension of latent class modeling mentioned in Sect. 6.3.5.

The use of Bayes nets in educational assessment combines ideas from two fields: psychometrics, as an extension of latent class modeling (Sect. 6.3.5), and knowledge engineering, as an application of context-sensitive model construction for probability-based reasoning (Breese et al. 1994). Bayes nets are used in fields including troubleshooting, medical diagnosis, and intelligence analysis that tackle complicated evolving inquiries in real time by assembling probability modules that reflect evidentiary characteristics of both static and dynamic situations (see Pourret et al. (2008) for examples).

In educational assessment, an analyst can construct probability modules for a distribution for current knowledge about a possibly vector-valued latent variable $\boldsymbol{\theta}$ and link functions $p(X_k \mid \boldsymbol{\theta})$ for classes of situations k that provide evidence about $\boldsymbol{\theta}$ which can occur or can be caused to occur.

In the Hydrive simulation system for troubleshooting the hydraulics systems of the F-15 aircraft, one such class was encountering a situation in which it was possible to carry out a space-splitting move in a canopy problem (Mislevy & Gitomer 1995). The X was whether the student's next sequence of actions was in fact space-splitting, or instead consistent with serial elimination, remove-and-replace, redundant, or irrelevant. The $\boldsymbol{\theta}$ parents were space-splitting strategic knowledge, canopy-system knowledge, and knowledge of the relevant troubleshooting procedures. Other examples of this approach appear in the Andes intelligent tutoring system for physics (VanLehn et al. 2005) and for assessing students' inquiry skills in the immersive environment of the Taiga Park computer game (Shute 2011). Behrens and DiCerbo (2014) note the promise of a Bayes net approach to synthesizing evidence about students' learning across multiple forms and contexts of activity in what they call "the digital ocean," continually updating belief over time as students participate in various learning activities in order to support learning as it occurs in various ways and at various times, as opposed to obtaining information only in focused assessment occasions.

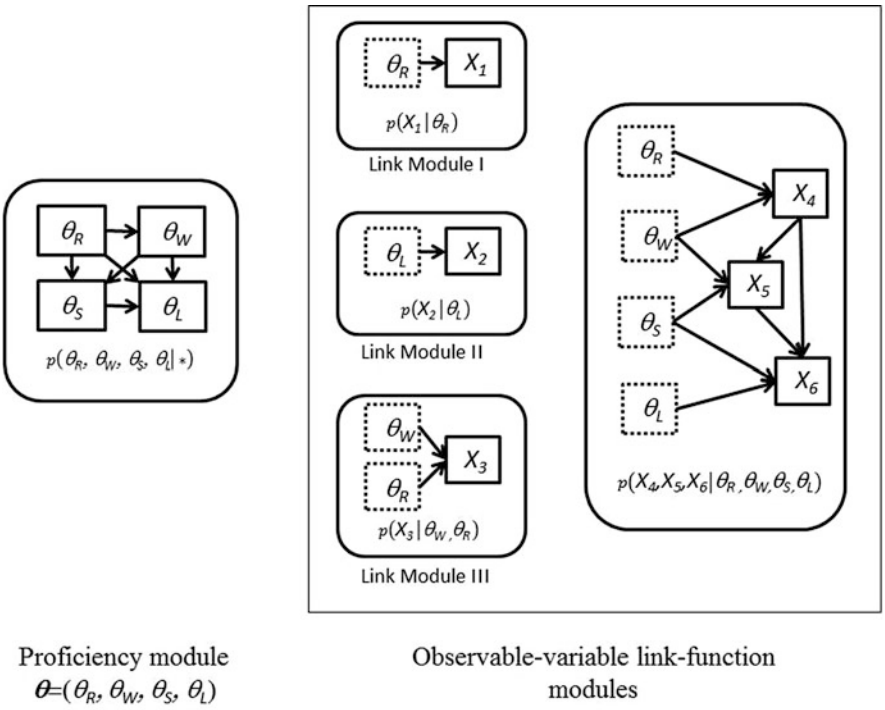


Fig. 6.2 Probability-model modules for use in assessment with dynamic model construction

Figure 6.2 is an illustration with a four-dimensional latent-variable proficiency fragment and four link functions that can be combined with it in appropriate situations to make observations and update belief about $\boldsymbol{\theta}$ using Bayes Theorem. The link functions correspond to situation types a student can encounter, perhaps multiple times, while acting in the task environment. When such a situation occurs, either pre-constructed or arising through the interaction, the specific actions of the student are evaluated to produce values of instances of the observable variable types $\boldsymbol{X}$ that are modeled as depending on certain aspects of proficiency $\boldsymbol{\theta}$ through the link functions. (Note that an observational variable that is input may be the possibly-multidimensional result of complex or multistage evaluations produced by data analytic techniques such as those discussed in Chaps. 9–11, 13 and 14. We will say more about this in the next section.) In Situation Type III in the figure, one observable variable X_3 obtains, and the corresponding link function is $p(X_3|\theta_W, \theta_S)$. When such a situation occurs, either by the examiner's actions (e.g., presenting a prespecified task) or is recognized as it arises in ongoing activity (e.g., by agents in a digital task environment like Hydrive), an instance of Link Function III is attached to the current distribution for $\boldsymbol{\theta}$ and it is updated via Bayes Theorem in light of the observed value for this instance of X_3.

These applications might rightly be called instances of computational psychometrics, but they can be seen as extensions of concepts and methods from the psychometric literature sketched above. The knowledge-based model construction extends IRT-based CAT. The Bayes nets fragments derive from latent class and CDMs, and the features that activate them correspond to rows in an implicit Q-matrix. The conditional probability matrices in the fragments can be parsimoniously structured around multidimensional IRT models for categorical responses (Almond et al. 2015), and their application over classes of situations rather than specific instances builds on work on the analogous problem in IRT (Geerlings et al. 2011).

6.4.4 *Interfaces Between Psychometric and Data-Analytic Models*

The psychometric models discussed above are models for observable variables, or $\boldsymbol{X}$s, given unobservable person variables $\boldsymbol{\theta}$. Attention focuses on "explaining" relationships among $\boldsymbol{X}$s within and between persons in terms of more parsimonious, psychologically-motivated aspects of persons' capabilities. In other words, $\boldsymbol{X}$s are but a starting point, arising in social contexts, motivated by psychological perspectives, and shaped by available technologies.

As discussed in previous chapters, most current educational assessments have historically provided relatively simple and relatively sparse $\boldsymbol{X}$s. With objectively-scored tasks such as multiple choice items, the assessment environments and response spaces were constrained to make the reasoning from performance to X simple and straightforward. With more complex, interactive tasks such as open-ended designs and investigations, human raters produced similarly simple Xs, such as ratings on a small number of scoring scales. Much sophisticated reasoning may lay beneath such ratings, but it is largely hidden in the raters' heads (Chap. 2). In other words, $\boldsymbol{X}$s are but a end point of more complex activities of design and analysis that focus on "understanding" relationships among features of performance within and between persons in terms of more parsimonious, psychologically-motivated aspects of persons' cognitive processing—processing generated by the more fundamental capabilities.

A prime motivation of computational psychometrics is dealing with complex performance situations, for more psychologically sophisticated views of students' capabilities. In simulation-based tasks, for example, quite voluminous data can be captured about every detail of a student's actions—every time-stamped mouse click, every key stroke, every utterance to every other participant, and in some cases heart rates, eye fixations, and physical locations. Reasoning from micro-features of performance to higher-order inferences about students' capabilities can become quite challenging.

There are two primary ways the psychometric concepts and methods discussed above can make technical contributions in the chain of reasoning (see Chaps. 1–4

for additional discussion). The first is the notion of a "psychometric backbone"—persistent variables, $\boldsymbol{\theta}$, for characterizing higher-level psychological constructs, which are posited to hold meaning for understanding performance across multiple real or hypothetical specific situations (e.g., tasks, specific content, different settings, different pathways in an interactive environment). The other is serving in stages in evidence-identification, in a multilayered chain of reasoning from low-level data to indicators of psychologically-relevant patterns of performance to serve as high-level Xs that enter in to psychometric models.

Techniques from data mining, learning analytics, and machine learning have proven valuable in discovering and detecting evidentiarily-relevant patterns in low-level data from complex performances. Several are discussed in other chapters of the present volume. Psychometric models can also play this evidence identification role. For example, (Scalise 2017) used small, local Bayes nets to produce intermediate summaries of students' strategies across clusters of related lower-level features of their activity in a simulation-based inquiry task. The results became Xs in a psychometric backbone, specifically in that application, a higher-level two-dimensional Rasch model with θs for inquiry and explanation.

Whether evidence-identification within performances uses psychometric models or other techniques to produce evidentiary summaries of task performances, there are benefits to using a psychometric backbone when one wishes to synthesize evidence either within tasks or across tasks when different students are administered different tasks or follow evidentiarily different paths within tasks. The connection of theories of the domain and performance with the statistical model contributes to the *validity* argument. The probability framework for aspects of proficiency addresses *reliability* in the form of posterior distributions at any point in time. Extending the core ideas of IRT DIF, the ability to test link functions for interactions with student background variables is a rigorous tool for aspects of *fairness*. Mapping the evidence from performances into a common $\boldsymbol{\theta}$ framework provides for *comparability*, not only of what, but for quantifying how much evidence is obtained. Hierarchical extensions of the models across different contexts supports investigations of the *generalizability* of extrapolation inferences from performances in given contexts, as an extension of Cronbach's generalizability-theory framework.

6.4.5 Exploratory and Confirmatory Psychometric Modeling

While psychometrics concerns the use of quantitative models to study psychological phenomena, we may highlight a distinction that has arisen in previous sections as it bears on the use of psychometrics in more complex assessments. It is the distinction between confirmatory modeling and exploratory modeling, and their interplay in practical work (Jones & Thissen 2007). Purely confirmatory modeling begins with a conception of psychological attributes, expresses them as latent variables θ, and provides link functions $p(x|\theta)$ that give conditional probabilities of observable variables x. The expected relationships among observable variables are

thus generated by the psychometric model. Purely exploratory modeling begins with data xs and estimates a structure that more parsimoniously characterizes patterns in the data. The revealed patterns can hold clues to psychologically meaningful understanding of the phenomena of interest.

Among the psychometric models mentioned above that can be used in either mode are factor analysis, latent class analysis, and, when Q-matrices are estimated from data, CDMs. Other psychometric models that are mainly exploratory include cluster analysis, which produces groups of cases with similar patterns of observable variables, and multidimensional scaling, which locates cases as points in a lower-dimensional space; a more comprehensive discussion of these and many other models is included in Jones and Thissen (2007). In the former case, the groups may reveal social or psychological similarities; in the latter, regions in the space may suggest meaningful social or psychological dimensions that characterize persons or situations. We note in passing that factor analysis, cluster analysis, and latent class analysis were innovative data mining techniques for the big data of their day. They remain as part of the data analyst's tool kit, and their central ideas have been further extended and generalized in more recent methods such as latent semantic analysis, Bayesian network modeling, and support vector machines (Romero et al. 2010).

6.4.6 Network Psychometrics

Recently, in the psychometric literature an alternative representation of psychometric constructs has been proposed: Instead of attributing the relationships between observed variables as arising from the underlying latent variable related to all the observed variables, the set of observed variables is seen as a network with causal interactions among these variables. The field of network psychometrics (Marsman et al. 2018; Epskamp et al. 2018) aims to estimate network models from psychological datasets in an attempt to map out the complex interplay between psychological, biological, sociological, and other components. These network models view observed variables as nodes and the strength of conditional association between two variables after controlling for all other variables as edges. Higher-level parameters—i.e., latent variables—that may characterize persons and situations determine expected associations in the network.

In certain cases the equations for the joint probability distributions of the observable variables given underlying parameters in a network model are the same as those arising from an IRT model with person and task variables. Such models represent a synergy among two psychometric developments discussed in Sect. 6.3, namely, IRT models and social network analysis.

6.5 Conclusion

Over the course of more than a century, psychometricians have developed a body of concepts and methods for making principled psychological inferences about individuals from observations of their performances in collections of specific situations. The concepts and methods originated within the context of limited data types, technologies of the time for capturing and analyzing data, and the trait and behavioral psychological perspectives; They were instantiated in forms attuned to this context.

These psychometric concepts and models, however, are grounded on more general principles that continue to hold value even as psychology and digital technologies advance, enabling new forms of assessment, richer and more complex data, and more sophisticated inferential targets. It has been our goal to review developments in psychometrics that not only describe their particular forms, but bring out the underlying principles and point to ways they can be gainfully employed in the next generations of assessments—psychometrics reconceived, extended, and integrated with techniques and insights from the several other fields that are required jointly.

References

Almond, R. G., Mislevy, R. J., Steinberg, L. S., Yan, D., & Williamson, D. M. (2015). *Bayesian networks in educational assessment*. New York: Springer.

Bartholomew, D. J. (1980). Factor analysis for categorical data. *Journal of the Royal Statistical Society. Series B (Methodological), 42*(3), 293–321.

Behrens, J. T., & DiCerbo, K. E. (2014). Harnessing the currents of the digital ocean. In J. Larusson & B. White (Eds.), *Learning analytics* (pp. 39–60). New York, NY: Springer.

Birnbaum, A. (1968). Some latent trait models and their use in inferring an examinee's ability. In F. M. Lord & M. R. Novick, *Statistical theories of mental test scores* (pp. 395–479). Oxford, UK: Addison-Wesley.

Bolt, D. M., & Lall, V. F. (2003). Estimation of compensatory and noncompensatory multidimensional item response models using Markov chain Monte Carlo. *Applied Psychological Measurement, 27*(6), 395–414.

Bradlow, E. T., Wainer, H., & Wang, X. (1999). A Bayesian random effects model for testlets. *Psychometrika, 64*(2), 153–168.

Breese, J. S., Goldman, R. P., & Wellman, M. P. (1994). Introduction to the special section on knowledge-based construction of probabilistic and decision models. *IEEE Transactions on Systems, Man, and Cybernetics*, *24*(11), 1577–1579.

Byrne, B. M., Shavelson, R. J., & Muthén, B. (1989). Testing for the equivalence of factor covariance and mean structures: The issue of partial measurement invariance. *Psychological Bulletin, 105*(3), 456.

Chen, Y., Culpepper, S. A., Chen, Y., & Douglas, J. (2018). Bayesian estimation of the DINA Q matrix. *Psychometrika, 83*(1), 89–108.

Cronbach, L., Gleser, G., Nanda, H., & Rajaratnam, N. (1972). *The dependability of behavioral measurements: Theory of generalizability for scores and profiles.* New York: Wiley.

De Boeck, P., & Partchev, I. (2012). IRTrees: Tree-based item response models of the GLMM family. *Journal of Statistical Software, 48*, 1–28.

De Boeck, P., & Wilson, M. (2004). *Explanatory item response models*. New York: Springer.
De la Torre, J. (2011). The generalized DINA model framework. *Psychometrika, 76*(2), 179–199.
Embretson, S. E. (1991). A multidimensional latent trait model for measuring learning and change. *Psychometrika, 56*(3), 495–515.
Epskamp, S., Maris, G., Waldorp, L. J., & Borsboom, D. (2018). Network psychometrics. In P. Irwing, D. Hughes, & T. Booth (Eds.), *The Wiley handbook of psychometric testing.* New York: Elsevier.
Fischer, G. H. (1973). The linear logistic test model as an instrument in educational research. *Acta Psychologica, 37*(6), 359–374.
Fox, J.-P. (2010). *Bayesian item response modeling: Theory and applications*. New York: Springer Science & Business Media.
Gao, X., Shavelson, R. J., & Baxter, G. P. (1994). Generalizability of large-scale performance assessments in science: Promises and problems. *Applied Measurement in Education, 7*(4), 323–342.
Geerlings, H., Glas, C. A., & van der Linden, W. J. (2011). Modeling rule-based item generation. *Psychometrika, 76*(2), 337.
Gelman, A., Stern, H. S., Carlin, J. B., Dunson, D. B., Vehtari, A., & Rubin, D. B. (2013). *Bayesian data analysis*. New York: Chapman and Hall/CRC.
Gulliksen, H. (2013). *Theory of mental tests*. New York: Routledge.
Henson, R. A., Templin, J. L., & Willse, J. T. (2009). Defining a family of cognitive diagnosis models using log-linear models with latent variables. *Psychometrika, 74*(2), 191.
Jansen, B. R., & van der Maas, H. L. (2002). The development of children's rule use on the balance scale task. *Journal of Experimental Child Psychology, 81*(4), 383–416.
Jones, L. V., & Thissen, D. (2007). A history and overview of psychometrics. In C. R. Rao & S. Sinharay (Eds.), *Handbook of statistics* (Vol. 26, pp. 1–27). New York: Elsevier.
Joreskog, K. G., Sorbom, D., & Magidson, J. (1979). *Advances in factor analysis and structural equation models.* New York, NY: New York University Press.
Junker, B. W., & Sijtsma, K. (2001). Cognitive assessment models with few assumptions, and connections with nonparametric item response theory. *Applied Psychological Measurement, 25*(3), 258–272.
Lazarsfeld, P. F. (1959). Latent structure analysis. *Psychology: A Study of a Science, 3*, 476–543.
Levy, R. (2009). The rise of Markov chain Monte Carlo estimation for psychometric modeling. *Journal of Probability and Statistics, 2009*, Article ID 537139.
Levy, R., & Mislevy, R. J. (2016). *Bayesian psychometric modeling*. New York: Chapman and Hall/CRC.
Little, R. J., & Rubin, D. B. (2014). *Statistical analysis with missing data*. Hoboken, NJ: Wiley.
Liu, J., Xu, G., & Ying, Z. (2012). Data-driven learning of Q-matrix. *Applied Psychological Measurement, 36*(7), 548–564.
Lord, F. M. (1980). *Applications of item response theory to practical testing problems*. Hillsdale, NJ: Lawrence Erlbaum Associates.
Lord, F. M., & Novick, M. R. (1968). *Statistical theories of mental test scores*. Oxford, UK: Addison-Wesley.
Luo, Y., & Jiao, H. (2018). Using the Stan program for Bayesian item response theory. *Educational and Psychological Measurement, 78*(3), 384–408.
Magis, D., Béland, S., Tuerlinckx, F., & De Boeck, P. (2010). A general framework and an R package for the detection of dichotomous differential item functioning. *Behavior Research Methods, 42*(3), 847–862.
Marsman, M., Borsboom, D., Kruis, J., Epskamp, S., van Bork, R. van, Waldorp, L., … Maris, G. (2018). An introduction to network psychometrics: Relating Ising network models to item response theory models. *Multivariate Behavioral Research, 53*(1), 15–35.
Mellenbergh, G. J. (1989). Item bias and item response theory. *International Journal of Educational Research, 13*(2), 127–143.
Meredith, W. (1993). Measurement invariance, factor analysis and factorial invariance. *Psychometrika, 58*(4), 525–543.

Messick, S. (1994). The interplay of evidence and consequences in the validation of performance assessments. *Educational Researcher, 23*(2), 13–23.

Millsap, R. E. (2012). *Statistical approaches to measurement invariance*. New York: Routledge.

Mislevy, R. J. (2016). Missing responses in item response modeling. In W. J. van der Linden (Ed.), *Handbook of item response theory, volume two: Statistical tools* (pp. 171–194). Boca Raton, FL: Chapman Hall/CRC Press.

Mislevy, R. J., Behrens, J. T., DiCerbo, K. E., & Levy, R. (2012). Design and discovery in educational assessment: Evidence-centered design, psychometrics, and educational data mining. *Journal of Educational Data Mining, 4*(1), 11–48.

Mislevy, R. J., & Gitomer, D. H. (1995). The role of probability-based inference in an intelligent tutoring system. *User Modeling and User-Adapted Interaction, 5*(3), 253–282.

Mislevy, R. J., & Wu, P.-K. (1996). *Missing responses and IRT ability estimation: Omits, choice, time limits, and adaptive testing*. Tech. Rep. No. RR-96-30-ONR. Princeton, NJ: Educational Testing Service.

Murphy, K. P. (2002). *Dynamic Bayesian networks: Representation, inference and learning*. Unpublished doctoral dissertation, University of California at Berkeley.

National Research Council. (2001). *Knowing what students know: The science and design of educational assessment*. Washington, D.C.: National Academies Press.

Patz, R. J., & Junker, B. W. (1999). A straightforward approach to Markov chain Monte Carlo methods for item response models. *Journal of Educational and Behavioral Statistics, 24*(2), 146–178.

Pourret, O., Naïm, P., & Marcot, B. (2008). *Bayesian networks: A practical guide to applications*. John Wiley & Sons.

Rabe-Hesketh, S., & Skrondal, A. (2004). *Generalized latent variable modeling: Multilevel, longitudinal, and structural equation models*. New York: Chapman and Hall/CRC.

Raju, N. S. (1988). The area between two item characteristic curves. *Psychometrika, 53*(4), 495–502.

Rasch, G. (1960). *Probabilistic models for some intelligence and achievement tests*. Copenhagen: Danish Institute for Educational Research.

Reckase, M. D. (2009). Multidimensional item response theory models. In M. D. Reckase (Ed.), *Multidimensional item response theory* (pp. 79–112). New York: Springer.

Romero, C., Ventura, S., Pechenizkiy, M., & Baker, R. S. (2010). *Handbook of educational data mining*. New York: Chapman and Hall/CRC press.

Rupp, A., Templin, J., & Henson, R. (2010). *Diagnostic assessment: Theory, methods, and applications*. New York: Guilford.

Scalise, K. (2017). Hybrid measurement models for technology-enhanced assessments through mIRT-bayes. *International Journal of Statistics and Probability, 6*(3), 168.

Shute, V. J. (2011). Stealth assessment in computer-based games to support learning. In J. D. Fletcher & S. Tobias (Eds.), *Computer games and instruction* (pp. 503–524). Charlotte, NC: Information Age Press.

Spearman, C. (1904). "General Intelligence," objectively determined and measured. *The American Journal of Psychology, 15*(2), 201–292.

Swaminathan, H., & Rogers, H. J. (1990). Detecting differential item functioning using logistic regression procedures. *Journal of Educational Measurement, 27*(4), 361–370.

Takane, Y., & De Leeuw, J. (1987). On the relationship between item response theory and factor analysis of discretized variables. *Psychometrika, 52*(3), 393–408.

Tatsuoka, K. K. (1983). Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement, 20*(4), 345–354.

Thissen, D., Steinberg, L., & Wainer, H. (1988). Use of item response theory in the study of group differences in trace lines. In H. Wainer & H. I. Braun (Eds.), *Test validity* (pp. 147–169). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Tucker, L. R. (1955). The objective definition of simple structure in linear factor analysis. *Psychometrika, 20*(3), 209–225.

VanLehn, K., Lynch, C., Schulze, K., Shapiro, J. A., Shelby, R., Taylor, L.,... Wintersgill, M. (2005). The Andes physics tutoring system: Lessons learned. *International Journal of Artificial Intelligence in Education, 15*(3), 147–204.

von Davier, M. (2008). A general diagnostic model applied to language testing data. *British Journal of Mathematical and Statistical Psychology, 61*(2), 287–307.

Wang, X., Berger, J. O., Burdick, D. S., et al. (2013). Bayesian analysis of dynamic item response models in educational testing. *The Annals of Applied Statistics, 7*(1), 126–153.

Wasserman, S., & Faust, K. (1994). *Social network analysis: Methods and applications*. Cambridge, UK: Cambridge University Press.

Way, W. D., Ansley, T. N., & Forsyth, R. A. (1988). The comparative effects of compensatory and noncompensatory two-dimensional data on unidimensional IRT estimates. *Applied Psychological Measurement, 12*(3), 239–252.

Chapter 7
Bayesian Inference in Large-Scale Computational Psychometrics

Gunter Maris, Timo Bechger, and Maarten Marsman

Abstract This chapter provides an introduction to Bayesian inference using Markov Chain Monte Carlo (MCMC) methods. We focus on two popular MCMC methods: Metropolis-Hastings and the Gibbs sampler. A Metropolis-Hastings algorithm developed by Marsman et al. (Sci Rep 5:9050, 1–7, 2015) will be used to illustrate how MCMC can be done for a wide range of models in computational statistics.

7.1 Introduction

Most latent variable models in psychometrics have in common that the distribution of the observed variables is intractable, either because it involves an integral that cannot be solved analytically, or because it involves a sum over all possible response patterns. As a consequence, direct likelihood inference is not possible, and some kind of approximation is needed.

Note: The R or Python codes can be found at the GitHub repository of this book: https://github.com/jgbrainstorm/computational_psychometrics.

G. Maris (✉)
Metior Consulting, Arnhem, The Netherlands

University of Amsterdam, Amsterdam, The Netherlands
e-mail: gunter@metior.consulting

T. Bechger
Metior Consulting, Arnhem, The Netherlands
e-mail: timo@metior.consulting

M. Marsman
University of Amsterdam, Amsterdam, The Netherlands
e-mail: m.marsman@uva.nl

© Springer Nature Switzerland AG 2021

A. A. von Davier et al. (eds.), *Computational Psychometrics: New Methodologies for a New Generation of Digital Learning and Assessment*, Methodology of Educational Measurement and Assessment, https://doi.org/10.1007/978-3-030-74394-9_7

Over the past 25 years, Bayesian inference has increased tremendously in popularity, in no small part because of increased computational power (Green et al. 2015). Bayesian statistics assumes that parameters are random variables, and hence the distribution of the parameters conditionally on the observations is well defined through Bayes Theorem:

$$p(\text{parameters}|\text{data}) = \frac{p(\text{data}|\text{parameters})\, p(\text{parameters})}{p(\text{data})}$$

The marginal distribution of the parameters is commonly called the *prior* distribution to distinguish it from the conditional distribution which is called the *posterior* distribution.

If we can simulate from the posterior distribution, Monte Carlo integration (Geweke 1989) can be used to approximate intractable integrals. This entails nothing more than replacing an expectation by a sample mean. Unfortunately, the posterior distribution itself is typically as intractable as is the distribution of the observed variables and getting an independent and identically distributed (iid) sample from it is not possible.

A number of "tricks" have been developed to overcome this problem. The first one is *Data Augmentation* (Tanner & Wong 1987). For latent variable models this approach comes naturally as the distribution of the observed variables is defined as a marginal of the joint distribution of observed and latent variables. Since the latent variables render the observed variables conditionally independent, the joint distribution of observed and latent variables is much easier to work with. For models belonging to the class of *latent response models* (Maris 1995), which includes most of the models encountered in psychometrics, an additional layer of latent variables is built into the model formulation. For instance, in the normal ogive model the distribution of an observed binary variable conditionally on a latent continuous variable is a Gaussian integral, and the joint distribution of observed binary variables, latent responses (i.e., the thing inside the Gaussian integral) and latent variables is well defined. Data Augmentation is a powerful tool, but rarely sufficient to allow for directly simulating from the posterior distribution.

A second "trick" consists of giving up the idea of an *iid* sample. Specifically, we settle for a dependent sample from the posterior distribution as this still allows for Monte Carlo integration. Typically a Markov chain (Gilks et al. 1995) is constructed for which the posterior distribution is the invariant distribution. In a Markov chain, the $(t + 1)$-th sample only depends on the t-th sample and not on any other sample. Starting from anywhere, the Markov chain is guaranteed to converge to simulating from the posterior distribution (Tierney 1994). Most Bayesian methods today are of this type, and the field is called *Markov Chain Monte Carlo (MCMC)*. General purpose algorithms such as the Metropolis-Hastings algorithm (Metropolis et al. 1953; Hastings 1970) allow in principle to simulate from any multivariate distribution. In practice however, they are rarely used as such.

The third "trick", which is also a general purpose Markov chain algorithm, is the Gibbs sampler (Geman & Geman 1984; Casella & George 1992). The Gibbs

sampler reduces simulating from a high dimensional distribution to simulating from a number of low (typically one) dimensional distributions.

Most MCMC methods in actual use employ all three of these tricks. One of the first successful MCMC methods is the Gibbs sampler for the normal ogive model of Albert (1992) that relies on two layers of augmented variables (latent responses and latent variables). The algorithm capitalizes on unique properties of the normal distribution. Over the years, a number of general purpose algorithms and computer programs were developed, such as BUGS (Gilks et al. 1994), JAGS (Plummer et al. 2003), or JASP (Love et al. 2019), that can deal with a large class of models. For a general introduction to the field of MCMC the interested reader is referred to the monographs by Gelman et al. (2014) and Robert and Casella (2013), and for a specific introduction to MCMC for psychometrics to the monograph by Fox (2010).

As data sets become bigger and models become more complex MCMC becomes a challenge. The posterior standard deviation goes to zero at a rate of one divided by the square root of the sample size. At the same time, the computational cost of every iteration increases with sample size, as there are more parameters to be estimated. Hence, starting from the same point, more iterations are needed to locate the area where the posterior mass is concentrated and every iteration becomes more expensive as the sample size grows. For most MCMC algorithms the autocorrelation (which determines the step size) does not depend on sample size (Liu et al. 1994), but for some it actually increases (Papadimitropoulou 2013). Below, we'll introduce an MCMC algorithm for which autocorrelation actually *de*creases with sample size, making it a scalable solution.

In the following we illustrate how these three tricks can be put to good use to build an MCMC algorithm that scales well and can handle a large class of models. It will become clear, that MCMC is a bit of an art as well as a craft (van Dyk & Meng 2001). We can cover only a small part of the vast literature on MCMC and, for illustrations sake, we focus on an MCMC algorithm due to Marsman et al. (2015) which we believe is an efficient and general device for problems in computational psychometrics.

7.2 Gibbs Sampling

The *Gibbs sampler* (Geman & Geman 1984) is a "divide-and-conquer" strategy to sample from a multivariate posterior by sampling from the posterior distributions of individual parameters given the data and the current value of all other parameters. Gibbs sampling is popular because these so-called *conditional posterior (CP)* distributions[1] are usually easier to simulate from.

[1] Sometimes called *full-conditional distributions*, these are posteriors given all other parameters are known.

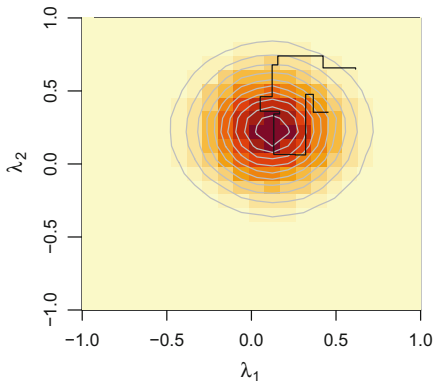


Fig. 7.1 Ten iterations of a Gibbs sampler with two parameters drawn on a contour plot of the posterior. Iterations start at the top right

We begin with a short introduction to the Gibbs sampler, explaining *how* it works, and *why*.

7.2.1 About the Gibbs Sampler

Let $\Lambda = (\Lambda_1, \ldots, \Lambda_m)$, $m \geq 2$, denote a vector of parameters.[2] The Gibbs sampler generates a sample $\lambda^{(0)}, \lambda^{(1)}, \ldots$ from the Markov chain that converges to the multivariate posterior $f(\lambda|\mathbf{y})$, where $\mathbf{y}$ are the data. Starting with an initial value $\lambda^{(0)}$ individual parameters are sampled independently and in any convenient order from their CP distributions in each iteration.

As illustrated in Fig. 7.1, a Gibbs sampler "travels" through the support of the posterior along horizontal and vertical paths. The initial estimate may have low posterior probability and it is customary to discard the first n values to give the sampler time to reach the posterior support. This is called the burn-in. Once inside the posterior support, the Gibbs sampler should be able to go anywhere, irrespective of the point of departure. This is true for the majority of applications and it is certainly true for the applications discussed here. Auto-correlation is inversely related to the step-size and should be low for the chain to move around.

A Gibbs sampler produces a sample from a Markov chain and, provided the sampler can reach all parts of the posterior support, convergence to the posterior is guaranteed if the posterior is its invariant distribution. This means that if $\lambda^{(0)}$ is drawn from the posterior, all subsequent values are also draws from the posterior. Suppose, for ease of presentation, that there are two parameters.[3] Their posterior density is

[2] We use subscripts to distinguish parameter vectors from scalars.

[3] The argument for the general case follows by mathematical induction.

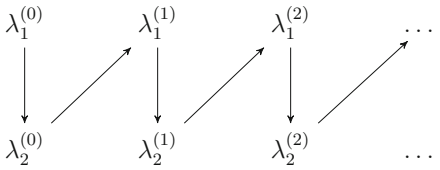


Fig. 7.2 Schematic picture of the Gibbs sampling procedure

$$f(\lambda_1, \lambda_2|\mathbf{y}) = f(\lambda_1|\lambda_2, \mathbf{y}) f(\lambda_2|\mathbf{y}) \quad .$$

To sample from this posterior, we draw $\lambda_2^{(1)}$ from the marginal posterior distribution and then $\lambda_1^{(1)}$ from the distribution conditional upon $\lambda_2^{(1)}$. Note that the latter is the CP distribution of parameter λ_2.

Suppose we set up a Markov chain to draw $\lambda_2^{(1)}$ from the marginal posterior distribution. We aim for a weak degree of dependence between subsequent samples. Specifically, we ensure that $\Lambda_2^{(1)}$ and $\Lambda_2^{(0)}$ are independent and identically distributed *conditional upon* $\Lambda_1^{(0)}$. That is, they are exchangeable

$$f(\lambda_2^{(0)}, \lambda_2^{(1)}|\mathbf{y}) = \int f(\lambda_2^{(0)}|\lambda_1^{(0)}, \mathbf{y}) f(\lambda_2^{(1)}|\lambda_1^{(0)}, \mathbf{y}) f(\lambda_1^{(0)}|\mathbf{y}) d\lambda_1^{(0)} \quad .$$

Exchangeability is a very weak form of statistical dependence (Ross 1996, 7.2), and hence comes with low autocorrelation, which in turn implies that Gibbs samplers move through the parameter space efficiently.

If we integrate $f(\lambda_2^{(0)}, \lambda_2^{(1)}|\mathbf{y})$ with respect to $\lambda_2^{(0)}$ (or $\lambda_2^{(1)}$), we see that $\Lambda_2^{(0)}$ and $\Lambda_2^{(1)}$ have the same marginal distribution. This distribution is the marginal posterior. It follows that

$$\begin{aligned} f(\lambda_2^{(1)}|\lambda_2^{(0)}, \mathbf{y}) &= \frac{f(\lambda_2^{(0)}, \lambda_2^{(1)}|\mathbf{y})}{f(\lambda_2^{(0)}|\mathbf{y})} \\ &= \int f(\lambda_2^{(1)}|\lambda_1^{(0)}, \mathbf{y}) \frac{f(\lambda_2^{(0)}|\lambda_1^{(0)}, \mathbf{y}) f(\lambda_1^{(0)}|\mathbf{y})}{f(\lambda_2^{(0)}|\mathbf{y})} d\lambda_1^{(0)} \\ &= \int f(\lambda_2^{(1)}|\lambda_1^{(0)}, \mathbf{y}) f(\lambda_1^{(0)}|\lambda_2^{(0)}, \mathbf{y}) d\lambda_1^{(0)}. \end{aligned}$$

To produce a value $\lambda_2^{(1)}$ from the posterior distribution we may use the method of composition as follows:

1. Draw $\lambda_2^{(0)}$ from the posterior.
2. Draw $\lambda_1^{(0)}$ from the CP $f(\lambda_1|\lambda_2^{(0)}, \mathbf{y})$.
3. Draw $\lambda_2^{(1)}$ from the CP $f(\lambda_2|\lambda_1^{(0)}, \mathbf{y})$.

This procedure is a Gibbs sampler starting with a draw from the posterior.

With $\lambda_2^{(1)}$ drawn from the marginal posterior, we then draw $\lambda_1^{(0)}$ from the CP $f(\lambda_1|\lambda_2^{(1)}, \mathbf{y})$ and repeat the process with $\lambda_2^{(1)}$ replacing $\lambda_2^{(0)}$, etc. Schematically, the sampling procedure may be depicted as in Fig. 7.2. It can be shown that these values are the realization of a Markov chain whose invariant distribution is, by construction, the posterior.

Let's look at a simple example to see what the CP distributions look like. The *Two-Parameter Logistic (2PL)* model is characterized by the following joint distribution of observed binary variables (X_i):

$$p(\mathbf{x}|\boldsymbol{\alpha}, \boldsymbol{\delta}) = \int_{\mathcal{R}} \prod_i \frac{\exp(x_i(\alpha_i\theta - \delta_i))}{1 + \exp(\alpha_i\theta - \delta_i)} f(\theta) d\theta$$

in which student ability θ is a latent variable, and α_i and δ_i are, respectively, the discrimination and difficulty parameter of item i (Birnbaum 1968). If we let $f(\delta_i)$ ($f(\alpha_i)$) denote the prior distribution of δ_i (α_i), the augmented posterior has the following form:

$$f(\boldsymbol{\theta}, \boldsymbol{\delta}, \boldsymbol{\alpha}|\mathbf{x}) \propto \prod_p \left(\prod_i \frac{\exp(x_i(\theta_p - \delta_i))}{1 + \exp(\theta - \delta_i)} f(\theta_p) \right) \prod_i f(\delta_i) f(\alpha_i)$$

The un-normalized CP distribution of ability θ_p is obtained by dropping everything from the augmented posterior that does not depend on it:

$$f(\theta_p|\boldsymbol{\theta}_{(p)}, \boldsymbol{\delta}, \boldsymbol{\alpha}, \mathbf{x}) \propto \prod_i \frac{\exp(x_i(\alpha_i\theta_p - \delta_i))}{1 + \exp(\alpha_i\theta_p - \delta_i)} f(\theta_p) \tag{7.1}$$

Note that the CP distribution for δ_i and α_i has the exact same form. In fact, as we'll see later on, this type of distribution shows up in many latent variable models. Furthermore, simulating from the CP distributions of the latent variables can be done in parallel. The same goes for the α_i and δ_i, which contributes to scalability. Unfortunately, there is no known way to directly sample from this distribution. We therefore turn to the Metropolis-Hastings algorithm next, as it allows for sampling from a different (simpler) distribution and making up for the mistake.

7.2.2 The Metropolis-Hastings Algorithm

The Metropolis-Hastings (MH) algorithm is a general purpose method to simulate from intractable distributions. It is a remarkable algorithm in that it simulates from the wrong distribution (called a *proposal distribution*) and then effectively makes up for the mistake by sometimes not accepting the sample from the proposal

distribution. If the proposal value is not accepted, the new value is the same as the current value.

If X_t is a draw from the distribution we want to sample from, Y_{t+1} a draw from the proposal distribution, and the binary random variable A_{t+1} denotes whether to accept the proposal or not we obtain the following representation[4]

$$X_{t+1} \sim \begin{cases} X_t & \text{if } A_{t+1} = 0 \\ Y_{t+1} & \text{if } A_{t+1} = 1 \end{cases} \sim X_t$$

The remarkable thing about the MH algorithm is that it is always (under some very mild regularity conditions, Tierney 1994) possible to specify the acceptance probability $p(A_{t+1} = 1|x_t, y_{t+1})$ such that the resulting Markov chain has X_t as its invariant distribution.

As it is instructive to see how this comes about we derive the result in some more detail for a real valued random variable. The distribution of X_{t+1} produced by the algorithm can be expressed as follows:

$$\begin{aligned} P(X_{t+1} \leq x) = F(x) = & \int_{\mathcal{R}} \int_{-\infty}^{x} \pi(s,t) g(s|t) f(t) ds dt \\ & + \int_{-\infty}^{x} \int_{\mathcal{R}} (1 - \pi(s,t)) g(s|t) f(t) ds dt \qquad (7.2) \\ & = \int_{\mathcal{R}} \int_{-\infty}^{x} \pi(s,t) g(s|t) f(t) ds dt \\ & + F(x) - \int_{-\infty}^{x} \int_{\mathcal{R}} \pi(s,t) g(s|t) f(t) ds dt \qquad (7.3) \end{aligned}$$

where g denotes our proposal distribution, f the target distribution, and π the acceptance probability. The equation encodes that we want the distribution after the update to be F if the distribution of the current value is also F. If we now cancel the $F(x)$ from both left and right hand side, and rename the variables in the second equation we obtain:

$$\int_{\mathcal{R}} \int_{-\infty}^{x} \pi(s,t) g(s|t) f(t) ds dt = \int_{\mathcal{R}} \int_{-\infty}^{x} \pi(t,s) g(t|s) f(s) ds dt$$

Clearly, if the functions to be integrated are the same, so will the integrals be, and hence we obtain the following, so called, *detailed balance condition*:

$$\pi(s,t) g(s|t) f(t) = \pi(t,s) g(t|s) f(s)$$

[4] $X \sim Y$ denotes that the random variables X and Y are identically distributed.

This equation has many solutions for the function π, but the one which makes the probability of accepting the proposal largest is the following one:

$$\pi(s,t) = \min\left(1, \frac{g(t|s)f(s)}{g(s|t)f(t)}\right) \tag{7.4}$$

Note that there are no particular constraints on the proposal distribution; it could be anything. However, not all proposal distributions are born equal, and much of the art and craft of MH sampling lies in choosing efficient proposal distributions that closely resemble the target distribution, and are simple to simulate from. A recipe that we found particularly useful is provided by the sum-matched Metropolis-Hastings algorithm of Marsman et al. (2015). A brief introduction is provided in the next section. Details of the algorithm and R code can be found in the Appendix and the electronic supplement (located the at the book code repository).

7.2.3 The Sum-Matched Metropolis-Hastings Algorithm

Suppose we have a 2PL with known item parameters and we wish to produce a *plausible value* (Mislevy 1991) for a person with response pattern $\mathbf{x}$; that is, a sample from the posterior of ability. To this aim, we will sample from the CP distribution for ability in the 2PL in Eq. (7.1) using MH. The question is: How to build an effective proposal distribution?

The key insight is that the CP distribution of ability (7.1) can be re-written as:

$$f(\theta_p|\boldsymbol{\theta}_{(p)}, \boldsymbol{\delta}, \boldsymbol{\alpha}, \mathbf{x}) \propto \prod_i F_i(\theta_p)^{x_i} \left(1 - F_i(\theta_p)\right)^{1-x_i} f(\theta_p), \tag{7.5}$$

where each $F_i(\theta_p) = P(X_{pi} = 1|\theta_p, \alpha_i, \delta_i)$ is a logistic cumulative distribution function (cdf) with location $\delta_i\alpha_i^{-1}$ and (positive) scale parameter α_i^{-1} with density:

$$f_i(\theta_{pi}) = \frac{\alpha_i \exp(\alpha_i\theta_{pi} - \delta_i)}{(1 + \exp(\alpha_i\theta_{pi} - \delta_i))^2}$$

If α_i is negative we may simply recode the observations, as explained in the Appendix.

The *Sum-Matched Metropolis-Hastings (SM-MH)* algorithm uses this simple fact to generate a proposal value in the following way. We first simulate values $\theta_{p1}, \theta_{p2}, \ldots, \theta_{pn}$ from each of these logistic distributions, as well as from the distribution of the latent variable (which we'll call θ_{p0} with distribution f_0) independently. If we let $\Theta_{p(j)}$ indicate the j-th order statistic from this sample, we readily find it is distributed as:

$$f(\theta_{p(j)}) \propto \prod_{i \neq j} F_i(\theta_{p(j)})^{y_{ij}} (1 - F_i(\theta_{p(j)}))^{1-y_{ij}} f_j(\theta_{p(j)}), \tag{7.6}$$

where $y_{ij} = 1$ if $\theta_{pi} \leq \theta_{p(j)}$ and zero otherwise. In fact, we generate a collection of $n + 1$ proposal distributions, one for every order of the order statistic. Each has the right form but differs from the target. Either because $j \neq 0$ and the prior has traded place with one of the items, or because the observed response pattern, $\mathbf{x}$ differs from the generated one, $\mathbf{y}$. The MH algorithm is used to correct for this difference.

By definition, the sum $y_{+j} = \sum_{i \neq j} y_{ij}$ is the number of values smaller than $\theta_{p(j)}$ so that every proposal has a different sum. The SM-MH algorithm chooses the one whose sum corresponds to the sum of the observed response pattern, $x_+ = \sum_i x_i$. This should lead to a good proposal, and one for which the acceptance probability is extremely simple to compute as shown in the Appendix. Moreover, the differences between generated and observed response pattern become irrelevant, and the proposal gets better the more observations we have (Bechger et al. 2018). Specifically, the acceptance probability goes to one which means that autocorrelation decreases with sample size. Figure 7.3 illustrates this using simulated data.

Summarizing: The SM-HM algorithm provides a proposal that is simple to simulate from, it scales well, and the acceptance probability is easy to compute. The algorithm is tailor-made for particular kind of distribution. The examples in the next section illustrate that this kind of CP distributions show up in a wide range of models.

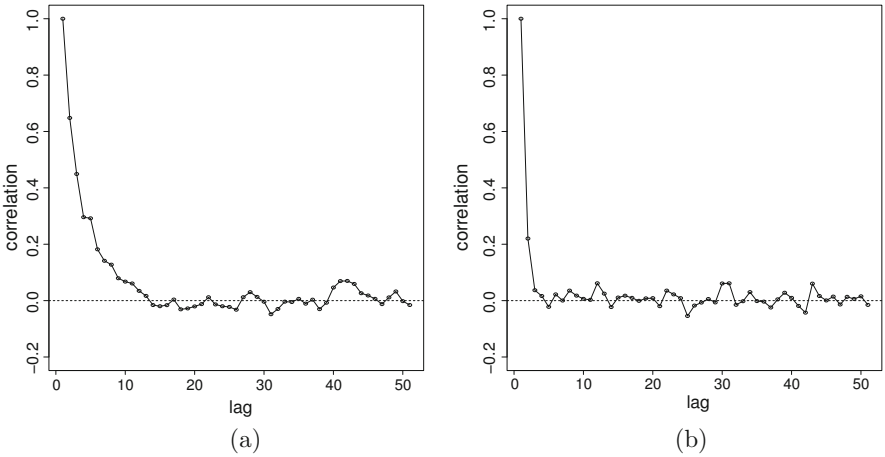


Fig. 7.3 Autocorrelations (lag 1 to 50) for a 2PL with small (left) and moderate (right) sample size. (**a**) $a_i \sim \mathcal{U}(0.1, 2.1)$, $n = 50$. (**b**) $a_i \sim \mathcal{U}(0.1, 2.1)$, $n = 5000$

7.3 Examples

In general, the SM-MH algorithm produces a dependent sample from a CP distribution of the form:

$$f(\eta|\mathbf{x}, \mathbf{a}, \mathbf{b}) \propto f(\eta, \mathbf{x}) = \prod_{i=1}^{n} F_i(\eta)^{x_i} \left(1 - F_i(\eta)\right)^{1-x_i} f(\eta), \tag{7.7}$$

where $\mathbf{x}$ is a vector of n binary observations, η is the parameter of interest with prior density $f(\eta)$, and

$$F_i(\eta) = P(X_i = 1|\eta, a_i, b_i) = \frac{\exp(a_i\eta + b_i)}{1 + \exp(a_i\eta + b_i)}$$

is a logistic cdf. In this section, we provide some examples to illustrate how the SM-MH algorithm can be used as a plug-and-play device to build Gibbs samplers. Throughout, our strategy will be to re-write a CP distribution in the form (7.7), with $\mathbf{x}$ the part of the data that provides information about the parameter of interest, η, while $\mathbf{a}$ and $\mathbf{b}$ summarize the values of the other parameters.

Note that the electronic supplement of this chapter provides more detail and the reader will find R code there to "try it at home".

7.3.1 Logistic Regression

We observe a binary outcome y_p, coded 0/1, and m predictor variables $w_{p1}, \ldots, w_{pm}$ for a sample of n_p persons. It is assumed that

$$P(Y_p = 1|w_{p1}, \ldots, w_{pm}) = \frac{e^{\sum_j \beta_j w_{pj}}}{1 + e^{\sum_j \beta_j w_{pj}}}$$

Our aim is to determine the posterior of the regression coefficients[5] $\boldsymbol{\beta}$. Their joint posterior is

$$p(\boldsymbol{\beta}|\mathbf{y}, \mathbf{w}) = \prod_p \left(\frac{e^{\sum_j \beta_j w_{pj}}}{1 + e^{\sum_j \beta_j w_{pj}}} \right)^{y_p} \left(1 - \frac{e^{\sum_j \beta_j w_{pj}}}{1 + e^{\sum_j \beta_j w_{pj}}} \right)^{1-y_p} \prod_j f(\beta_j)$$

Removing everything unrelated to β_j, we find its CP distribution:

[5] To introduce and intercept, just add a vector of ones as a predictor.

$$f(\beta_j|\mathbf{w},\mathbf{y}..) = \prod_p \left(\frac{e^{\beta_j w_{pj}+\sum_{k\neq j}\beta_k w_{pk}}}{1+e^{\beta_j w_{pj}+\sum_{k\neq j}\beta_k w_{pk}}}\right)^{y_p} \left(1-\frac{e^{\beta_j w_{pj}+\sum_{k\neq j}\beta_k w_{pk}}}{1+e^{\beta_j x_{pj}+\sum_{k\neq j}\beta_k w_{pk}}}\right)^{1-y_p} f(\beta_j)$$

where the dots represent the other coefficients. This CP has the form of Eq. (7.7) and we can use the SM-MH algorithm with $\eta = \beta_j$, $x_p = y_p$, and constants $a_p = w_{pj}$ and $b_p = \sum_{k\neq j}\beta_k w_{pk}$.

When the number of predictors is large, we could choose a Laplace(0,λ^{-1}) prior which would give us a *Bayesian Lasso* (Park & Casella 2008; Hastie et al. 2015). If we give λ^{-1} a Gamma(α,β) hyperprior, its CP distribution is Gamma($\alpha + m$,$\beta + \sum_j |\beta_j|$).

7.3.2 Multi-Dimensional IRT

Consider the following *multi-dimensional IRT model*

$$P(Y_{pi} = 1|\boldsymbol{\theta}_p, \boldsymbol{\lambda}, \boldsymbol{\delta}) = \frac{\exp\left(\sum_{k=1}^{n_k}\lambda_k c_{ik}\theta_{pk} + \delta_i\right)}{1+\exp\left(\sum_{k=1}^{n_k}\lambda_k c_{ik}\theta_{pk} + \delta_i\right)} \tag{7.8}$$

where the latent ability $\boldsymbol{\theta}_p = (\theta_{p1}, \ldots, \theta_{pn_k})$ is multidimensional with $1 \leq n_k < n_i$ the number of latent dimensions. We consider a confirmatory model where the c_{ij} are known contrasts (or *factor-loadings*) and our main interest is in the parameters $\lambda_k > 0$ which give the importance of the kth factor defined by $\mathbf{c}_{.k} = (c_{1k}, \ldots, c_{n_i k})$. The details for an exploratory model are left as an exercise for the reader.

A Gibbs sampler for this model can be build in the same way as the one for the 2PL. The difference is that it is slightly more involved to calculate the constants; **a**, and **b**. Summarizing,

$$\eta = \delta_i \Rightarrow \begin{cases} \mathbf{x} = \mathbf{y}_{.i} \\ \mathbf{a} = \mathbf{1}_{n_p} \\ b_p = \sum_j \lambda_j c_{ij}\theta_{pj} \end{cases} \qquad \eta = \theta_{pj} \Rightarrow \begin{cases} \mathbf{x} = \mathbf{y}_{p.} \\ \mathbf{a} = \lambda_j \mathbf{c}_{.j} \\ b_i = \delta_i + \sum_{h\neq j}\lambda_h c_{ih}\theta_{ph} \end{cases}$$

When we sample λ_j, the observations **x** comprise the whole data matrix stored as a vector. If we denote by (ip) an index that runs over all $n_i \times n_p$ observations in the same order in which the data matrix is stored as a vector, we obtain:

$$a_{(ip)} = c_{ij}\theta_{pj}, \text{ and } \quad b_{(ip)} = \delta_i + \sum_{h\neq j}\lambda_h c_{ih}\theta_{ph} \tag{7.9}$$

Note that $a_{(ip)} = 0$ for all i where $c_{ij} = 0$; that is, for all items that do not load on factor j.

A special case of the model is a *constrained* bi-factor model (Holzinger & Swineford 1937) where the items form clusters such that within each cluster all items have the same factor-loadings. The first dimension is the general factor and $\mathbf{c}_{i1} = 1$, for all items i. The other dimensions correspond to the cluster-factors with $c_{ij} = 1$ if item i belongs to cluster j and $c_{ij} = 0$ otherwise, for $j > 1$. The loading of factor j is λ_j for all items i where $c_{ij} = 1$. For illustration, we use this model to analyze the responses of 500 first year psychology students to 240 personality items (Dolan et al. 2009) developed to measure individual differences in five personality traits known as "the big five": *neuroticism*, *extraversion*, *agreeableness*, *openness to experience* and *conscientiousness*. These traits form the clusters. The data were dichotomized with one if a respondent felt a description applied to her, and a zero if not.

It took about 1 min to do a thousand iterations of the Gibbs sampler. The lowest acceptance probabilities are round 0.80 for the cluster factor-loadings. As illustrated in Fig. 7.4, the constrained bi-factor model reproduces,[6] in a highly parsimonious way, the blocked pattern typically seen in the observed correlations; a pattern that cannot be reproduced by a 2PL or even a 3PL.

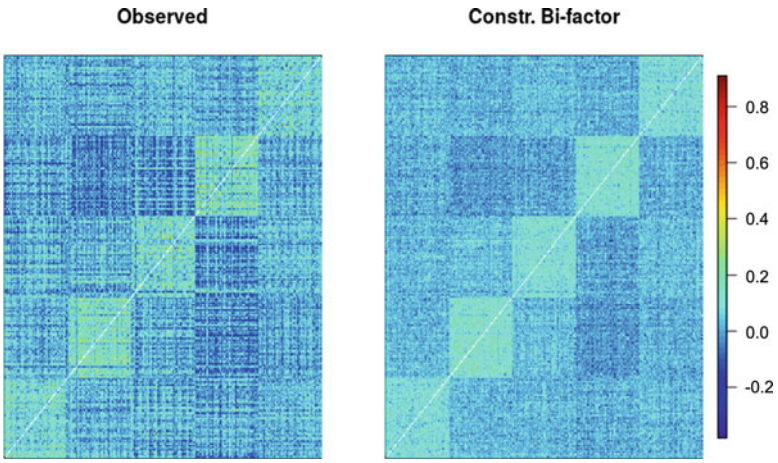


Fig. 7.4 Image plots of observed correlations and correlations replicated under a constrained bi-factor model

[6] We generated posterior predictive data to calculate the correlations.

7.3.3 An "Inter-Leaved" Gibbs Sampler for the Ising Model

We have independent observations on a set of dependent binary random variables Y_i, distributed as an Ising (Lenz 1920) model:

$$P(\mathbf{y}|\boldsymbol{\Sigma}, \boldsymbol{\mu}) = \frac{\exp\left(\frac{1}{2}\sum_i \sum_j \sigma_{ij} y_i y_j + \sum_i \mu_i y_i\right)}{Z(\boldsymbol{\Sigma}, \boldsymbol{\mu})} \tag{7.10}$$

where the σ_{ij} are the entries of a symmetric connectivity matrix $\boldsymbol{\Sigma}$ and express the strength of pair-wise dependencies; i.e., variables Y_i and Y_j are independent conditional upon all others when $\sigma_{ij} = 0$. The parameters μ_i function in the same way as item difficulty (or easiness) in IRT models. The normalizing constant $Z = Z(\boldsymbol{\Sigma}, \boldsymbol{\mu})$ is called the partition function. The partition function is highly intractable as it can only be computed by enumeration over all possible binary response vectors.

In educational measurement, the Ising model is recognized as an exponential family model for a network of binary variables involving their main effects and pairwise interactions (Marsman et al. 2018). The variables typically form a dense network with every variable (positively) related to many others which implies that the connectivity matrix can be approximated well by a low-rank matrix. This enticed Marsman et al. (2015) to consider a *low-rank Ising model* - an Ising model where the rank of the connectivity matrix is set in advance to be much smaller than the number of variables.

Their Gibbs sampler is based on the observation that a rank-d Ising model is equivalent to a d-dimensional IRT model with a specific (prior) distribution of ability. Specifically, if the connectivity matrix $\boldsymbol{\Sigma}$ is of rank d, with λ_k its k-th eigenvalue, and c_{ik} the loading of the i-th item on the k-th dimension we find:

$$\begin{aligned} p(\mathbf{y}) = &\int_{\mathcal{R}^d} \prod_i \frac{\exp\left(y_i\left(\sum_{k=1}^{n_k} \lambda_k c_{ik}\theta_{pk} + \mu_i\right)\right)}{1 + \exp\left(\sum_{k=1}^{n_k} \lambda_k c_{ik}\theta_{pk} + \mu_i\right)} \\ &\times \left(\prod_i 1 + \exp\left(\sum_{k=1}^{n_k} \lambda_k c_{ik}\theta_{pk} + \mu_i\right)\right) \frac{\exp(-\boldsymbol{\theta}^T\boldsymbol{\theta})}{Z\pi^{N/2}} d\boldsymbol{\theta} \end{aligned} \tag{7.11}$$

where we directly recognize the multidimensional IRT model (Eq. (7.8)), and a somewhat peculiar distribution of ability, which Hinton et al. (2006) have dubbed the *complementary prior distribution*. This relation has been discovered independently in as diverse fields as statistical physics (Kac 1968), machine learning (Hinton et al. 2006), spatial statistics (Besag 1975), biometrics (McCullagh 1994), statistics (Wermuth & Lauritzen 1990; Olkin et al. 1961), and indeed psychometrics (Anderson & Yu 2007), and implies that being able to estimate the Ising model has impact throughout the statistical sciences.

A key insight is that the complementary prior distribution makes the posterior distribution of ability particularly simple. First, the θ_k are independent conditionally

on the observed variables, and second, they are all normally distributed. These properties make generating plausible values trivial. Posterior distributions of item parameters, however, are intractable, as they involve the partition function, which cannot be evaluated. This is different for a *regular* multidimensional IRT model, where the situation is precisely reversed.

Marsman et al. (2015) propose to combine the simplicity of generating plausible values from the Ising model with the simplicity of sampling from the posterior distribution of item parameters from a regular multidimensional IRT model in an interleaved approximate Bayesian Gibbs sampler, which they refer to as *full data information estimation*. Theoretical work by Marsman et al. (2016) implies that the distribution of plausible values converges monotonically in expected Kullback-Leibler divergence, for any prior distribution. As a direct consequence, the approximate Bayesian Gibbs sampler converges to an exact Gibbs sampler as sample size increases. Combined with the increase in efficiency (decrease in autocorrelation) as sample size increases, and the ease of parallelization, this makes the full data information Gibbs sampler an attractive approach to estimating Ising models.

To illustrate the full data information estimation procedure, we consider a simulated example for a model introduced by Kac (1968). The network is a sequence of nodes on a line with exponentially decreasing interaction strength ($\sigma_{ij} = \exp(-\gamma|i-j|)$) between them. Even though the model has only a single parameter, it's eigenvectors are wave functions of increasing frequency and the connectivity matrix is of full rank. Figure 7.5 gives the current state of a rank 9 model in iteration 100 with 100K observations on 30 variables. Obviously, averaging over multiple samples from our Markov chain would improve the recovery of the factor loadings. However, even a single sample does a good job at recovering the factor loadings. The last two panels on the second row both contain the factor loadings on the 9-th dimension. Without knowledge of the true loadings it would have been difficult to guess that they are more than random clutter. From the first 5 dimensions you could guess the pattern, however.

7.3.4 An "Interleaved" Gibbs Sampler for Restricted Boltzmann Machines

A Restricted Boltzmann Machine (RBM, Smolensky 1986) is nothing but an Ising model on a bipartite graph. A bipartite graph is a graph with two layers (called hidden and observed) such that there are only connections between nodes from different layers. Being an Ising model, we could use the algorithm from the last section to do inference on RBMs. However, as we'll see we don't need to introduce extra continuous latent variables.

Formally, the RBM is characterized by the following distribution:

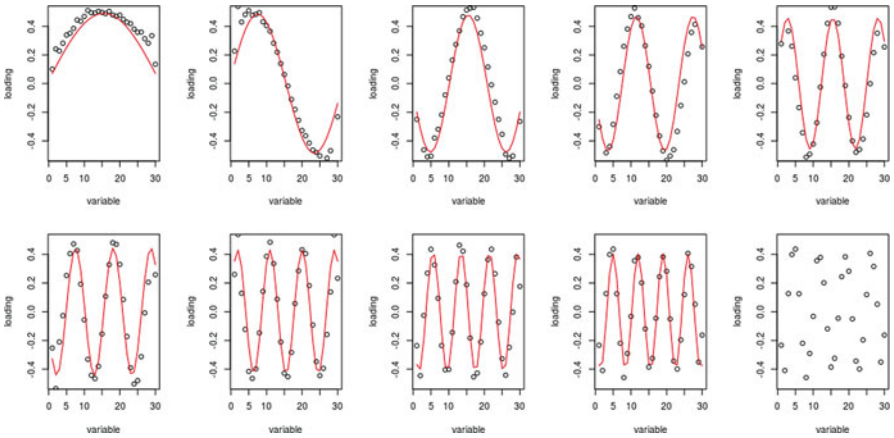


Fig. 7.5 True (red line) and estimated (black circles) loadings on 9 factors for the Kac exponential model on a line. The 10-th panel is the same as the 9-th one but without the true loadings

$$P(\mathbf{x}, \mathbf{y}|\boldsymbol{\Sigma}, \boldsymbol{\mu}, \boldsymbol{\nu}) = \frac{\exp\left(\frac{1}{2}\sum_i \sum_j \sigma_{ij} x_i y_j + \sum_i \mu_i x_i + \sum_j \nu_j y_j\right)}{Z(\boldsymbol{\Sigma}, \boldsymbol{\mu}, \boldsymbol{\nu})} \tag{7.12}$$

such that the distribution of the observed variables ($\mathbf{x}$) is

$$P(\mathbf{x}|\boldsymbol{\Sigma}, \boldsymbol{\mu}, \boldsymbol{\nu}) = \frac{\exp\left(\sum_i \mu_i x_i\right) \prod_j \left(1 + \exp\left(\nu_j + \sum_i \sigma_{ij} x_i\right)\right)}{Z(\boldsymbol{\Sigma}, \boldsymbol{\mu}, \boldsymbol{\nu})} \tag{7.13}$$

and conditionally on the latent variables we obtain

$$p(\mathbf{x}|\mathbf{y}, \boldsymbol{\Sigma}, \boldsymbol{\mu}, \boldsymbol{\nu}) = \prod_i \frac{\exp\left(x_i \left[\mu_i + \sum_j \sigma_{ij} y_j\right]\right)}{1 + \exp\left[\mu_i + \sum_j \sigma_{ij} y_j\right)}$$

It is immediately clear that using full data information estimation we get CP distributions of the same form as we've seen throughout the examples once again.

In much the same way as RBMs can be estimated, we can estimate latent tree models, deep Boltzmann machines, and many more of the popular models in machine learning.

7.4 Discussion

Bayesian inference and the MCMC algorithms that go with it are an indispensable tool for any researcher in modern day computational psychometrics. The founda-

tions were laid many decades ago in the statistical physics community and have since spread across the statistical sciences. We have offered a gentle introduction to some of the key concepts with special attention for psychometric applications (at scale).

As the examples show, there is a particular kind of distribution you need to simulate from that keeps popping up. For this particular distribution we have introduced an efficient simulation algorithm, originally due to Marsman et al. (2015), that has favourable statistical properties, and illustrated its use in a wide range of models. Specifically, since autocorrelation decreases with sample size and most parts of the algorithm can be computed in parallel, the algorithm is scalable. For distributions that are *doubly intractable* (Murray et al. 2012) we have introduced the idea of full data information estimation, also due to Marsman et al. (2015), which allows for approximate Bayesian inference, with the guarantee that the approximation improves with sample size.

It will be clear that, this chapter does no justice to the vast and growing literature on Bayesian inference and MCMC. A choice was made to introduce key concepts and illustrate with a particular algorithm that MCMC is an art and a craft rather than a recipe you can blindly follow. We hope that the chapter provides a novice to the field will have enough general understanding to further explore the literature and at the same time has a concrete tool in his or her toolbox with which real world problems can be dealt with. An R package[7] will be made available to this effect.

Some 25 years ago, when personal computers became widely available, it became feasible to use MCMC to estimate moderately complex models. Today's cloud computing services have unleashed unprecedented computational power which extends the complexity of models and the size of data sets that can be dealt with enormously. Combined with the fusing of machine learning and (computational) psychometrics, through the formal connection between the Ising model (aka Boltzmann machine) and MIRT models, the future (of Bayesian statistics) is bright.

7.5 Appendix

7.5.1 The SM-MH Algorithm

We wish to sample from a target posterior of the form:

$$f(\eta|\mathbf{x}) \propto f(\eta, \mathbf{x}) = \prod_{i=1}^{n} F_i(\eta)^{x_i} \, (1 - F_i(\eta))^{1-x_i} \, f(\eta),$$

[7] https://github.com/jessekps/jaBs.

where $\mathbf{x}$ is a vector of n binary observations, η is the parameter of interest with prior density $f(\eta)$, and

$$F_i(\eta) = P(X_i = 1|\eta, a_i, b_i) = \frac{\exp(a_i\eta + b_i)}{1 + \exp(a_i\eta + b_i)}$$

is a logistic cumulative distribution function with location $-b_i a_i^{-1}$ and (positive) scale parameter a_i^{-1}. Note that we suppress dependence on parameters when there is no risk of confusion.

We start by generating a proposal value from a proposal distribution based on the following innocuous lemma.

Lemma 1 *Consider a set of $n + 1$ independent random variables Z_r each with a potentially unique distribution function $F_r(z) = P(Z_r \leq z)$, where $r = 0, \ldots, n$. Define, for $r \neq j$, the random indicator variables*

$$Y_{rj} = (Z_r \leq Z_j) = \begin{cases} 1 & \text{if } Z_r \leq Z_j \\ 0 & \text{otherwise} \end{cases} .$$

Then, for $j \in \{0, \ldots, n\}$,

$$f_j(z_j, \mathbf{y}_j) = \prod_{r \neq j} F_r(z_j)^{y_{rj}} (1 - F_r(z_j))^{1-y_{rj}} f_j(z_j)$$

where $\mathbf{y}_j = (y_{rj})_{r \neq j}$.

Proof *Trivial.* □

The joint distribution $f_j(z_j, \mathbf{y}_j)$ is the proposal distribution and the lemma shows how to sample from it using *auxiliary variables* $Z_0, \ldots, Z_n$, where $Z_j \sim F_j$. We choose F_0 to be the prior distribution, and assume that F_j, for $j > 0$, is logistic so that $f_0(\eta, \mathbf{x})$ corresponds to the target. Proposal and target are "sum-matched" because we choose j such that $y_{+j} = \sum_{r \neq j} y_{rj} = \sum_i x_i = x_+$. In pseudo-code, this gives the following algorithm which we call the *Sum-Matched Approximate Bayes* or SM-AB algorithm.

Algorithm 1 SM-AB algorithm

for $r = 0$ **to** n **do**
  Sample: $z_r \sim F_r(z)$
Select the $(x_+ + 1)$th smallest of the z_r

Note that this algorithm is simple and computationally cheap. Cheap because only a partial sort is required to find the $(x_+ + 1)th$ smallest of set of numbers and computation time is of order n.

The SM-AB algorithm provides a sample from an approximate posterior. Adding a Metropolis-Hastings step to make up for this gives us the *Sum-Matched Metropolis-Hastings* or SM-MH algorithm.

Algorithm 2 SM-MH algorithm

Given $\eta^{(t)}$:

1. Draw an ability η^* and a response pattern $\mathbf{y}_j$ from $f_j(\eta^*, \mathbf{y}_j)$
2. Take

$$\eta^{(t+1)} = \begin{cases} \eta^* & \text{with probability } \pi(\eta^{(t)} \to \eta^*) = \min\{1, \alpha\} \\ \eta^{(t)} & \text{otherwise} \end{cases}$$

where

$$\alpha = \frac{f_0(\eta^*, \mathbf{x}) f_j(\eta^{(t)}, \mathbf{y}_j)}{f_0(\eta^{(t)}, \mathbf{x}) f_j(\eta^*, \mathbf{y}_j)}$$

The acceptance ratio

$$\alpha = \begin{cases} A_j \exp\left[(\eta^{(t+1)} - \eta^{(t)}) \left(\sum_{i \neq j} a_i x_i - \sum_{i \neq j} a_i y_{ij}\right)\right] & \text{if } j \neq 0 \\ \exp\left[(\eta^{(t+1)} - \eta^{(t)}) \left(\sum_i a_i x_i - \sum_i a_i y_{ij}\right)\right] & \text{if } j = 0 \end{cases} \tag{7.14}$$

with, for $j \neq 0$,

$$A_j = e^{(x_j - 1)[a_j(\eta^{(t+1)} - \eta^{(t)})]} \frac{1 + e^{a_j \eta^{(t+1)} + b_j}}{1 + e^{a_j \eta^{(t)} + b_j}} \frac{f_0(\eta^{(t+1)})}{f_0(\eta^{(t)})} \frac{F_0^{y_{0j}}(\eta^{(t)})[1 - F_0(\eta^{(t)})]^{1 - y_{0j}}}{F_0^{y_{0j}}(\eta^{(t+1)})[1 - F_0(\eta^{(t+1)})]^{1 - y_{0j}}}.$$

See Bechger et al. (2018). Note that only the second factor of A_j depends on the prior.

As sample size n increases, the approximate posterior converges to the posterior, the acceptance probability goes to one and the SM-AB and the SM-MH algorithm become equal. A formal proof can be found in Bechger et al. (2018).

7.5.2 *Dealing with Negative or Zero Scale Parameters*

The scaling parameters a^{-1} of the logistic distribution must be positive. Nevertheless, we will find situations where they are zero or negative. If $a_i = 0$, F_i becomes a constant with respect to η so that it can be dropped from the CP distribution. Thus,

we may delete from $\mathbf{x}$, $\mathbf{a}$ and $\mathbf{b}$, all entries i where $a_i = 0$. If some (or all) of the a_i are negative, we need to re-code the corresponding observations. To see this, note that

$$\begin{aligned} F_i(\eta)^{x_i}(1 - F_i(\eta))^{1-x_i} &= \left(\frac{e^{a_i\eta+b_i}}{1+e^{a_i\eta+b_i}}\right)^{x_i}\left(\frac{1}{1+e^{a_i\eta+b_i}}\right)^{1-x_i} \\ &= \left(\frac{e^{-(a_i\eta+b_i)}}{1+e^{-(a_i\eta+b_i)}}\right)^{1-x_i}\left(\frac{1}{1+e^{-(a_i\eta+b_i)}}\right)^{x_i} \end{aligned}$$

Thus, we re-code the observations *and* change the sign of a_i and b_i for all i where $a_i < 0$.

7.5.3 R Code

As a courtesy to the reader we provide a minimal set of R-scripts (R Core Team 2016) that read as pseudo-code for readers who program in other languages. Since R is one-based, the index of the prior is chosen to be $n + 1$, instead of 0.

7.5.3.1 Quickselect

At the time of writing, the quickselect algorithm is not yet implemented in R. A C++ implementation can be made available via the Rcpp-package (Eddelbuettel & Francois 2011).

```
#include <algorithm>
#include <functional>
#include <Rcpp.h>
using namespace Rcpp;
// [[Rcpp::plugins(cpp11)]]

// [[Rcpp::export]]
List qselect(NumericVector x, int k)
{
  double out;
  int iout;
  IntegerVector ivec(x.size());
  for (int i=0;i<ivec.size();i++) { ivec[i]=i;}
  std::nth_element(ivec.begin(),ivec.begin()+k-1,ivec.
     end(),[&](int i1, int i2) { return x[i1] < x[i2];
     });
  iout=ivec[k-1];
```

```
  out=x[iout];
  return List::create(Named("value", out),
                      Named("index", iout+1));
}
```

Note that the output is a list with the k-th smallest element and its original index; one-based for use in R.

7.5.3.2 The SM-MH Algorithm

The following is a straightforward implementation assuming a normal prior and index $n + 1$.

```
AB<-function(x,a,b,mu,sigma)
{
  n=length(x)
  z=rlogis(n,location=-b/a,scale=1/a)
  z=c(z,rnorm(1,mean=mu,sd=sigma))
  return(qselect(z,sum(x)+1))
}

MH<-function(x,a,b,current,mu,sigma)
{
  n=length(x)
  z=rlogis(n,location=-b/a,scale=1/a)
  z=c(z,rnorm(1,mean=mu,sd=sigma))
  abc=qselect(z,sum(x)+1)
  j=abc$index
  y=sapply(z[-j],function(x) 1*(x<=z[j]))
  if (j<(n+1))
  {
    c_1=(x[j]-1)*a[j]*(z[j]-current)
    c_2=log(1+exp(a[j]*z[j]+b[j]))
    c_3=log(1+exp(a[j]*current+b[j]))
    c_4=dnorm(z[j], mean=mu,sd=sigma,log = TRUE)
    c_5=dnorm(current,mean=mu,sd=sigma,log = TRUE)
    c_6=pnorm(current,mean=mu,sd=sigma,lower.tail=y[n],
       log.p = TRUE)
    c_7=pnorm(z[j],mean=mu,sd=sigma,lower.tail=y[n],log.
       p = TRUE)
    logA=c_1+c_2-c_3+c_4-c_5+c_6-c_7
    logalpha=logA+sum(a[-j]*(x[-j]-y[-n]))*(z[j]-current
       )
  }else
  {
```

```
    logalpha=sum(a*(x-y))*(z[j]-current)
  }

  new=current
  if (log(runif(1,0,1))<=logalpha) new=z[j]
  return(new)
}
```

7.5.3.3 A Gibbs Sampler for the 2PL

Let n_p be the number of persons, and n_I the number of items. The following script performs one iteration of a Gibbs sampler for the 2PL.

```
   ## MStep
theta=sapply(1:nP,function(p) MH(x[p,],alpha,delta,
   theta[p],mu.th,sigma.th))
delta=sapply(1:nI,function(i) MH(x[,i],rep(1,m),alpha[
   i]*theta,delta[i],0,2))
alpha=sapply(1:nI,function(i) MH(x[,i],theta,rep(delta
   [i],m),alpha[i],mu.al,sigma.al)

   ## identify
 s=sum(delta)/sum(alpha)
 m=1/sd(theta)
 delta = delta-s*alpha
 theta=m*(theta+s)
 alpha=alpha/m

  ## sample hyperparameters
 mu.th = rnorm(1,mean(theta),sigma.th/sqrt(nP))
 mu.al = rnorm(1,mean(alpha),sigma.al/sqrt(nI))
```

References

Albert, J. (1992). Bayesian estimation of normal ogive item response curves using Gibbs sampling. *Journal of Educational Statistics, 17*(3), 251–269.

Anderson, C. J., & Yu, H.-T. (2007). Log-multiplicative association models as item response models. *Psychometrika, 72*(1), 5–23.

Bechger, T., Maris, G., & Marsman, M. (2018). An asymptotically efficient Metropolis-Hastings sampler for Bayesian inference in large-scale educational measurement. Preprint, arXiv:1808.03947.

Besag, J. (1975). Statistical analysis of non-lattice data. *The Statistician, 24*(3), 179–195.

Birnbaum, A. (1968). Some latent trait models and their use in inferring an examinee's ability. In F. Lord & M. Novick (Eds.), *Statistical theories of mental test scores* (pp. 395–479). Reading, MA: Addison-Wesley.

Casella, G., & George, E. (1992). Explaining the Gibbs Sampler. *The American Statistician, 46*, 167–174.

Dolan, C., Oort, F., Stoel, R., & Wicherts, J. (2009). Testing measurement invariance in the target rotated multigroup exploratory factor model. *Structural Equation Modeling, 16*(2), 20–34.

Eddelbuettel, D., & Francois, R. (2011). Rcpp: Seamless R and C++ integration. *Journal of Statistical Software, 40*(8), 1–18.

Fox, J. (2010). *Bayesian item response modeling*. Springer.

Gelman, A., Carlin, J., Stern, H., Dunson, D., Vehtari, A., & Rubin, D. (2014). *Bayesian data analysis*(3rd ed.). Boca Raton, FL: CRC Press.

Geman, S., & Geman, D. (1984). Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 6*(6), 721–741.

Geweke, J. (1989). Bayesian inference in econometric models using Monte Carlo integration. *Econometrica: Journal of the Econometric Society, 57*(6), 1317–1339.

Gilks, W. R., Richardson, S., & Spiegelhalter, D. J. (1995). Introducing Markov chain Monte. In *Markov chain Monte Carlo in practice*.

Gilks, W. R., Thomas, A., & Spiegelhalter, D. J. (1994). A language and program for complex Bayesian modelling. *Journal of the Royal Statistical Society: Series D (The Statistician), 43*(1), 169–177.

Green, P. J., Łatuszyński, K., Pereyra, M., & Robert, C. P. (2015). Bayesian computation: A summary of the current state, and samples backwards and forwards. *Statistics and Computing, 25*(4), 835–862.

Hastie, T., Tibshirani, R., & Wainwright, M. (2015). *Statistical learning with sparsity: The lasso and generalizations*. CRC press.

Hastings, W. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika, 57*(1), 97–109.

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation, 18*(7), 1527–1554.

Holzinger, K., & Swineford, F. (1937). The bi-factor method. *Psychometrika, 2*(1), 41–54.

Kac, M. (1968). Statistical physics, phase transitions and superfluidity. *Brandeis University Summer Institute in Theoretical Physics, 1*, 241–305.

Lenz, W. (1920). Beiträge zum verstandnis der magnetischen eigenschaften in festen korpern. *Physikalische Zeitschrift, 21*, 613–615.

Liu, J. S., Wong, W. H., & Kong, A. (1994). Covariance structure of the Gibbs sampler with applications to the comparisons of estimators and augmentation schemes. *Biometrika, 81*(1), 27–40.

Love, J., Selker, R., Marsman, M., Jamil, T., Dropmann, D., Verhagen, J., ... others (2019). Jasp: Graphical statistical software for common statistical designs. *Journal of Statistical Software, 88*(2), 1–17.

Maris, E. (1995). Psychometric latent response models. *Psychometrika, 60*(4), 523–547.

Marsman, M., Borsboom, D., Kruis, J., Epskamp, S., van Bork, R. van, Waldorp, L., ... Maris, G. (2018). An introduction to network psychometrics: Relating Ising network models to item response theory models. *Multivariate Behavioral Research, 53*(1), 15–35.

Marsman, M., Maris, G., Bechger, T., & Glas, C. (2015). Bayesian inference for low-rank Ising networks. *Scientific Reports, 5*(9050), 1–7.

Marsman, M., Maris, G., Bechger, T. M., & Glas, C. A. W. (2016). What can we learn from plausible values? *Psychometrika, 81*(2), 274–289.

McCullagh, P. (1994). Exponential mixtures and quadratic exponential families. *Biometrika, 81*(4), 721–729.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., & Teller, A. (1953). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics, 21*(6), 1087–1092.

Mislevy, R. (1991). Randomization-based inference about latent variables from complex samples. *Psychometrika, 56*(2), 177–196.

Murray, I., Ghahramani, Z., & MacKay, D. (2012). Mcmc for doubly-intractable distributions. Preprint, arXiv:1206.6848.

Olkin, I., Tate, R. F., et al. (1961). Multivariate correlation models with mixed discrete and continuous variables. *The Annals of Mathematical Statistics, 32*(2), 448–465.

Papadimitropoulou, K. (2013). *A cautionary note on data augmentation algorithms*. Unpublished master's thesis, Universiteit Leiden.

Park, T., & Casella, G. (2008). The Bayesian lasso. *Journal of the American Statistical Association, 103*(482), 681–686.

Plummer, M., et al. (2003). Jags: A program for analysis of Bayesian graphical models using Gibbs sampling. In *Proceedings of the 3rd International Workshop on Distributed Statistical Computing* (Vol. 124, pp. 1–10).

R Core Team. (2016). R: A language and environment for statistical computing [Computer software manual]. Vienna, Austria: Retrieved from https://www.R-project.org/

Robert, C., & Casella, G. (2013). *Monte Carlo statistical methods*. Springer Science & Business Media.

Ross, S. (1996). *Stochastic processes*. Wiley.

Smolensky, P. (1986). *Information processing in dynamical systems: Foundations of harmony theory*. Tech. Rep., Colorado Univ at Boulder Dept of Computer Science.

Tanner, M. A., & Wong, W. H. (1987). The calculation of posterior distributions by data augmentation. *Journal of the American statistical Association, 82*(398), 528–540.

Tierney, L. (1994). Markov chains for exploring posterior distributions. *The Annals of Statistics, 22*(4), 1701–1728.

van Dyk, D., & Meng, X. (2001). The art of data augmentation. *Journal of Computational and Graphical Statistics, 10*, 1–50.

Wermuth, N., & Lauritzen, S. L. (1990). On substantive research: Hypothesis, conditional independence graphs and Graphical Chain Models. *Journal of the Royal Society. Series B (Methodological), 52*, 21–50.

Chapter 8
A Data Science Perspective on Computational Psychometrics

Jiangang Hao and Robert J. Mislevy

Abstract Digitally based learning and assessment systems generate large volumes of complex process data. The next generation psychometricians need to acquire new data science skills to meet the data challenge. In this chapter, we summarize data science skills and identify the subset that psychometricians need to prioritize. We introduce an evidence identification centered data design (EICDD) process during the task design, as an important way to address the data challenges from digitally based assessments. We describe some specific data techniques to parse and process complex process data with example codes in Python programming language. We also outline the general methodological strategies when dealing with process data from digitally based assessments.

8.1 Introduction

Digitally Based Assessments (DBAs) enable the capture of test takers' response process information at finer time granularity than traditional forms of assessment, and these rich process data can provide new opportunities for validating assessments, improving measurement precision, revealing response patterns/styles, uncovering group difference (fairness), detecting test security breaches, identifying new constructs, and providing feedback to learners and other stakeholders (Ercikan & Pellegrino, 2017; Mislevy et al., 2014). But these potential benefits do not come for free. The significantly increased volume, velocity, and variety of data pose new challenges to psychometricians for handling, analyzing, and interpreting the

The R or Python codes can be found at the GitHub repository of this book: https://github.com/jgbrainstorm/computational_psychometrics

J. Hao (✉) · R. J. Mislevy
Educational Testing Service, Princeton, NJ, USA
e-mail: jhao@ets.org; rmislevy@umd.edu

© Springer Nature Switzerland AG 2021

A. A. von Davier et al. (eds.), *Computational Psychometrics: New Methodologies for a New Generation of Digital Learning and Assessment*, Methodology of Educational Measurement and Assessment, https://doi.org/10.1007/978-3-030-74394-9_8

data to materialize their value. Data science is *an interdisciplinary field that uses scientific methods, processes, algorithms, and systems to extract knowledge and insights from data in various forms, both structured and unstructured* (Wikipedia contributors, 2018). Some techniques and practices used in data science could and should be adopted into the toolkit of next generation psychometrics (e.g., computational psychometrics, as suggested in this book) to help address the data challenges accompanying DBAs.

Broadly speaking, challenges concerning data can be put into two categories. The first is more on the engineering side, concerning capturing, hosting, transferring and manipulating data with proper software tools and techniques. The second is more substantive, concerning methods for making sense of data to realize their potential evidentiary value for intended interpretations and uses. The engineering side of handling data is generally less domain-dependent, while making sense of data usually depends heavily on the nature and intended use of the data in a specific discipline. Though the capabilities needed to address these two types of challenges may appear different, in practice they often overlap substantially. In recent years, some new professions have emerged to provide personnel and technological solutions to meet the needs of big data. For example, professionals with job titles of data architect and data engineer are usually expected to design and build enterprise data architectures and software infrastructures to host, transfer, and manage large-scale data. Those with a job title of data scientist are often expected to make sense of the data and materialize their value to support business.[1] The current chapter is not intended to provide a text on big-data engineering or architecting (interested readers may consult relevant volumes such as Kleppmann, 2017). In the current chapter, we introduce some data science techniques and modeling strategies in the context of DBAs to help psychometricians become better prepared to work with the increasingly big and complex data from next-generation assessments.[2]

The chapter is organized as follows. In Sect. 8.2, we propose that many data challenges can be mitigated through careful design of data early in design. To this end, we introduce an Evidence Identification-Centered Data Design (EICDD) process for use with the Evidence Centered Design (ECD; Almond, Steinberg, & Mislevy, 2002; Mislevy, Steinberg, & Almond, 2003) framework for DBAs. In Sect. 8.3, we introduce some data techniques and examples for manipulating structured and unstructured process data from DBAs. In Sect. 8.4, we introduce some methodological strategies towards modeling and interpreting complex response process data from DBAs. Finally, in Sect. 8.5, we summarize the chapter.

[1] These job classifications are not very strictly delineated in practice, given the overlapping skills needed to solve data-related problems.

[2] Some performance assessments use video, audio or other motion tracking mechanism (e.g., eye tracking) to capture test takers' performances. Specialized techniques are needed to process the video, audio or other motion tracking data, and there are established disciplines, such as computer vision and computational linguistics, for those topics. A discussion of these techniques is beyond the scope of this chapter. We focus primarily on the telemetry data that records test takers' interactions with the assessment.

8.2 Evidence Identification-Centered Data Design

In many fields, a data scientist often starts with data that have arisen coincidentally for purposes other than the ones the scientist has in mind. Neither their structure nor their contents may be optimal for the scientist's purpose. In an assessment, however, as it is often the case that how to use the data later (e.g., scoring and analytics) needs to be foreseen to some degree, analysts and psychometricians actually can and often should influence the data design early in the design process. Before delving into specific techniques to deal with messy data, it will be worth well to see whether some of the data challenges can be avoided if some careful considerations about data have been incorporated into the design process.

Evidence centered design (ECD) provides a general structure for designing complex assessments and spells out a methodological framework to relate the evidence and constructs of interest. One of the core processes in implementing ECD with DBAs is to identify prospective elements of the evidence contained in the telemetry data that record the interactions between the test taker and the assessment task. This includes in particular specifying the more generally defined ECD objects called Work Product Specifications, as well as the input and output data structures to be used by the Evidence Identification and Presentation processes. In practice, identifying the evidence is not always a technically trivial task, especially when data are poorly structured, have incorrect entries due to software bugs, or even fail to include the evidence that should be captured. One of the leading causes of these unfavorable situations in practice is that the people who are in charge of the data generation and capture process (usually the software programmers) are usually not those who will analyze them later (usually the data analysts and psychometricians). They have little incentive to design the data to be analysis friendly.

One possible way to avoid these unfavorable situations is to introduce an evidence identification-centered data design (EICDD) process for carrying out ECD when designing DBAs. There are at least two main components in an EICDD: (1) a data model that specifies the data structure and data type of each field, and (2) a set of standard operational procedures to guide what information should be stored into which field of the data model.

8.2.1 Data Model

A data model *is an abstract model that organizes elements of data and standardizes how they relate to one another and to the properties of real-world entities* (Wikipedia contributors, 2019, October 29). There are three main types of data models, namely, the relational, document, and graph-like models (Kleppmann, 2017). The relational data model, often showing up as tables in an SQL database, is better suited for the cases where the relationship among the attributes of data records is of the main concern in the application. The document data model, often showing

up as XML or JSON records in a NoSQL database, is better geared for cases where the main interest is within each data record, and there is not much interest in the cross-record relationship. The graph-like model, such as those implemented in a graph database, is better suited for data where the cross-record relationship is too complicated to describe with the relational model.

In principle, one can store the same data by following different data models, and it is not always easy to tell which one is significantly better than the others. In most learning and assessment applications, the natural unit of data is usually a learning/assessment session by a learner/test-taker, which can be naturally specified through a document data model. However, many psychometric analyses, such as equating and item calibration, depend on the relationship among many sessions, making the relational data model a good fit. In practice, the raw data, such as telemetry logs, are usually specified by a document model and are dumped into a data lake. Then, data processing, such as extraction and transformation, will be applied to the raw data. The processed data will be loaded either to another data lake or a data warehouse that is specified by relational models. Such a process is generally referred to as an ETL process. It is worth noting that many relational databases today allow the records in the form of XML/JSON strings. This makes the relational database itself a kind of data lake, though this may not be cost-competitive. In this case, the relational model maybe used to specify the relationship among the records that are specified by document models.

There are almost infinitely many ways to define specific data models. Some example data models for learning and assessment systems include the one specified under the Experience API (also known as xAPI or Tin Can API; Kevan & Ryan, 2016), the one specified under the Caliper Analytics (IMS Global Learning Consortium, 2015), and the document model for virtual performance assessments (VPAs) developed at Educational Testing Service (Hao et al., 2014, 2016). Figure 8.1 shows the structure of the ETS data model for VPAs.

8.2.2 Standard Operational Procedure

A data model specifies the structure and data type but does not tell what specific information from a particular task should be recorded into which field of the data model. In fact, given the almost infinite possibilities of assessment or learning tasks, this type of information is less likely to be specified through a set of fixed rules. As such, perhaps the best way to put some regularity into such a process is to establish some general guidelines and a set of "standard" operation procedures. In the context of DBAs, though they vary considerably, we can still offer some general guidelines about what events should be captured as follows.

Events in DBAs can be classified as test taker-triggered and system-triggered events. The test taker-triggered events can be further divided into two types. The first type involves the direct operations on elements designed for assessment, such as clicks on buttons, moves of game elements, typed responses, checked options, and

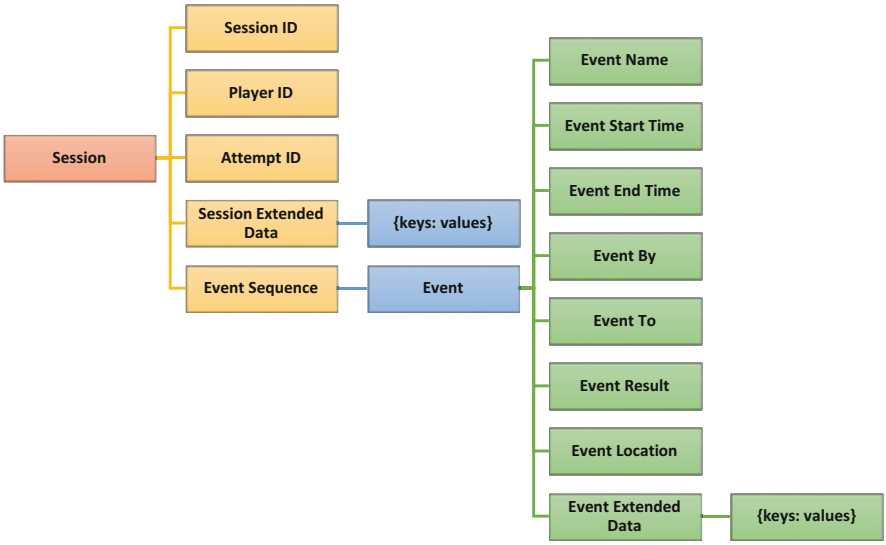


Fig. 8.1 ETS data model for virtual performance assessment tasks

verification of the DBA system status—quite specifically, instances of meaningful actions test takers take in situations with particular features. The entry for any such activity will include timing information and the identification of the elements of the DBA that are involved and, as needed, values of variables for relevant aspects of the state of the environment before and after the activity.

A challenge regarding the last of these is determining what constitutes a sufficient description of DBA states to associate with a given action. The same test taker's action can take different evidentiary meanings depending on prior actions or the state of the simulation. Note that *identifying what additional information is necessary to interpret the meaning of actions is a substantive, or scientific, question, not simply a data-capture question*. Thus, an appropriate set of attributes and parameters for each state needs to be established for each activity in collaboration between the members of the interdisciplinary design team for the DBAs. In early stages of development, finer grained and lower level events may need to be captured in the data structure so that patterns among them can be further explored to determine precisely which actions and system state variables are needed to produce the required evidence (e.g., DiCerbo et al., 2015; Halverson & Owen, 2014; Roberts et al., 2016). The subsequent operational version of the data structure may then need to contain only higher-level events computed on the fly if the required computation is straightforward to calculate.

The second type of test taker-triggered event involves activities that do not directly act on the DBA elements designed for assessment, such as clicks on the screen and typing on the keyboard. These events may be of use in developing behavioral analytics around test takers' engagement, motivation, and "gaming the

system" behaviors (Baker et al., 2004). In certain situations, this type of events may provide direct evidence of new constructs related to the process of the creation of the outcomes. For example, the keystroke events from the process of writing essays may be used to measure constructs related to writing fluency (Deane, 2014).

In addition to the test taker-triggered events, some system-triggered events should also be defined in a data specification. Some examples include events generated by the DBA itself, as in a change in a patient's condition occurring over time in a medical simulation independent of the test taker's actions; prespecified reports of the simulation system status, such as a "heartbeat" recording at regular intervals the values of a set of system-status variables that will be needed in subsequent processing to interpret test-taker actions (a set could include, for example, variables for simulation conditions such that certain configurations identify that "a particular dog is not barking;" Mislevy, 2018); and the "result" of a test-taker action can be a pointer to an accompanying more complex object that is itself a work product, such as an audio file or a graphics file containing a figure the test taker drew on a pressure-sensitive tablet.

Implementing the above guidelines in practice usually requires a synergy of different expertise, such as learning science, data science, psychometrics, and software development. When multiple actors are involved, it is always necessary to establish a set of operational procedures to specify the workflow and needed expertise in each step. Table 8.1 shows an example operational procedure developed at ETS to coordinate the process. Readers are encouraged to develop their own procedures in commensurate with their needs and available resources.

A careful design of data should be an integrated part of the solutions to the challenges that arise from the complex data of DBAs, but it often does not receive due attention in assessment design. Guiding the data design with evidence identification in mind, e.g., through an EICDD process as described above or similar ones, could help avoid many complications in later stages of evidence identification.

Table 8.1 Example of operational procedures that specify the steps and expertise in the instantiation of a data model in practice

X: participant XX: coordinator	Roles				
	Learning Scientist/ Assessment Developer	Psychometrician	Programmer	Data Scientist	Data analyst
Timeline					
Step 1: All-party meeting to brief the procedures	X	X	X	XX	X
Step 2: Specification of "Q matrix" for constructs and evidences	X				
Step 3: Specification of additional evidence for psychometric modeling need		X			
Step 4: Translate the evidences into operable task codes			X	X	XX
Step 5: Design log file structure and schema				X	
Step 6: Implement the evidence logging into the log file and tryout			X	X	XX
Step 7: Log file parsing and evidence extraction based on the tryout data				X	
Step 8: QA the log files to check whether the recovered evidence is sufficient. If not, restart from **Step 6**. If everything is good, proceed to next step	X	X		X	XX
Step 9: Finalize the data reduction pipeline and QA procedure				XX	X
Step 10: Implement the data reduction				X	XX

Hao and Mislevy (2018) introduced the term *Evidence Trace File* (ETF) to denote a data file that went through an EICDD (or similar) process, as distinguished from the common term *log file* that is widely used in VPAs for files that capture process events but have not gone through a purposive design procedure particular to the evidentiary needs and intended purposes of assessment. In a broader sense, we believe that a clear specification of what data are collected and how they are used can be critical for establishing the credibility and legitimacy of the assessment and mitigating test takers' concerns regarding fairness and privacy.

8.3 Data Techniques

Roughly speaking, there are two types of data processing, batch and stream processing. Batch processing is dealing with data that have already been collected for some time while stream processing is analyzing a continuous data stream generated or loaded in real time. In practice, the choice of batch or stream processing is dependent on the intended use and data size. For example, if the intended use depends on the cumulative information that are not generated at the same time, batch processing is generally considered. If the intended use is for providing immediate feedback based on the data up to the current time point, stream processing may be better suited. If the data size is very big and cannot be comfortably loaded into a modern computer's memory, one can turn the data into a data stream (e.g., many small chunks) and then apply stream processing techniques. Data from assessments, especially those from high-stake assessments, are usually processed in batch as many psychometric procedures, such equating and item response theory (IRT) modeling, are based on data that are not available simultaneously. But for data from adaptive learning or assessment systems that require real-time feedback or adaption, stream processing is obviously needed.

Different data infrastructure and ecosystems have been developed for different types of data processing. For example, in the big data paradigm as of the writing of this chapter, the ecosystems such as Apache Hadoop (https://hadoop.apache.org/) are primarily intended for batch processing, while ecosystems such as Apache Storm (http://storm.apache.org/) and Apache Kafka (https://kafka.apache.org/) are intended for stream processing. Some other ecosystems such as Apache Spark (https://spark.apache.org/) and Apache Flink (https://flink.apache.org/) are capable of handling both batch and stream processing. The details of these infrastructures are beyond the scope of this section and we refer the interested readers to the relevant tutorials on the above websites.

In the following subsections, we will introduce some basic data techniques for handling complex process data from some example DBAs. We will primarily focus on the batch processing, but also provide a lightweight introduction to stream processing using Python generator functions.

Table 8.2 Some Core Packages for Numerical Computation with Python

Package	Descriptions	Website
Numpy	Package for enabling numerical computation.	https://www.numpy.org
Scipy	Package for scientific computing based on numpy.	https://www.scipy.org
Pandas	Package for the data frame and tabular data manipulation	https://pandas.pydata.org
Scikit-learn	Package for machine learning.	https://scikit-learn.org
NLTK	Package for Natural Language Processing.	https://www.nltk.org
SpaCy	Package for Natural Language Processing.	https://spacy.io
Matplotlib	Package for plotting and visualization.	https://matplotlib.org
Seaborn	Package for plotting and visualization based on matplotlib.	https://seaborn.pydata.org
Plotly	Package for interactive visualization.	https://plotly.com

8.3.1 Python Ecosystem for Data Science

Traditional psychometric analysis usually starts with data in the form of a person-item table, with a convention of rows representing test takers and columns representing items. Each cell of the table is the response or the score of the response of a test taker with respect to an item as specified by the corresponding row and column. However, the process data from DBAs are generally not a regular response/score table. Instead, they often appear as a sequence of time-stamped events with different attributes and are either saved into standalone files or as entries in a database. These data can be stored either as an unstructured stack of strings (often due to some poor data engineering designs) or in a structured format such as XML or JSON. To identify evidence from these types of data, one needs to be able to manipulate the data at any granularity, which usually cannot be accomplished by GUI-based software (e.g., Excel). Programming using certain programming/scripting languages is generally necessary.

Python programming language is becoming popular among data scientists and software developers (Robinson, 2017). Differing from the R programming language primarily used for statistical analysis, Python supports a much wider range of applications, including website development, desktop software development, scientific computation, and even game development. Python comes with some basic functionality but relies heavily on external packages to perform almost all numerical computations. After a natural selection process over the last decade, a set of packages that provide fundamental computing capabilities has been widely accepted in the Python community. In Table 8.2, we list some of these core packages that are maintained by the Python community.

In addition to these backbone packages, the front end of the Python interpreter is also evolving. Probably the most user-friendly and interactive interface for Python at the moment is the Jupyter Notebook (Kluyver et al., 2016), or its new variant Jupyter Lab, which provides a browser-based interface for interactive data analysis.

The easiest way to get Python, the core packages, and Jupyter Notebook installed on computers is through the Anaconda suite (https://www.anaconda.com/download). Anaconda can be viewed as a management tool for Python packages and virtual environments, through which one can easily install/update Python packages from a well-maintained repository and create as well as manage virtual environments to run different versions of Python and packages. We encourage readers to explore the corresponding website to become familiar with installing the Python systems through anaconda. We assume in the subsequent discussion they have been installed.

8.3.2 Tabular and Sequential Data

To work with data, we need to have a certain data structure (e.g., an array) to hold the data and then apply different operations on it. Tabular data is probably the most familiar data structure to psychometricians. As we discussed in Sect. 8.2, data table is the typical instantiation of a relational data model that is well-geared for representing the inter-relationship among the data records. In Python, the Pandas package furnishes a data frame for holding tabular data. Pandas also comes with many convenient functions to facilitate computation and visualization based on the data frame (more details can be found from the Pandas documentation website at https://pandas.pydata.org/pandas-docs/stable/). On the other hand, the response process data from a DBA session are usually a sequence of time-stamped events with different attributes, which are usually stored as XML or JSON strings under a document data model.

In practical analyses, researchers often want to extract different features from the sequential data of each session and then analyze them across the sessions. As such, it is sometimes more convenient and efficient to restructure the sequential data from a session into a tabular data frame. In particular, indexing the rows of the data frame as the sequence number of events and the columns as different attributes of the events leads to a natural mapping from sequential data to tabular data. One can then stack the converted tabular data from different sessions together to create a tabular representation of many sessions. When doing such a mapping, there is a slightly increased redundancy, but this overhead is usually outweighed by increased convenience. Table 8.3 shows a schematic of a representation of sequential data in a tabular format.

Psychometricians have a long history of dealing with tabular data and have developed a wide range of techniques and capabilities. The tabular data psychometricians usually handle are organized in the so called wide format (e.g., each row is an observation/person, and each column is a response/score/feature). The tabular data converted from sequential data as described above are organized as a combination of wide format and long format (e.g., each row is an observation belonging to a specific category) due to the sequential nature of the data. Converting sequential data into appropriate tabular data frames is a crucial step in recasting the new data challenges into representations with which psychometricians are familiar. In the subsequent

Table 8.3 A schematic of the data frame converted from sequential data

Sequence number	Event time	Test taker ID	Session ID	Event name	Attribute 1	Attribute 2	...
1	Timestamp 1	A	1	...	...	...	...
2	Timestamp 2	A	1	...	...	...	
3	...	A	1	...	...	...	...
4	...	A	1	...	...		
5	...	B	2	...	...	...	...
6	...	B	2	...	...	...	...
7	...	B	2	...	...	...	...

```
[5/15/2013 2:17:26 PM] Session Start
[5/15/2013 2:17:26 PM] Leaving sequence: loadXML, moving forward.
[5/15/2013 2:17:30 PM] Player submitted name: Carl
[5/15/2013 2:17:30 PM] Leaving sequence: InputNameScreen, moving forward.
[5/15/2013 2:17:31 PM] Player submitted name: Carl
[5/15/2013 2:17:31 PM] Leaving sequence: startScreen, moving forward.
[5/15/2013 2:17:50 PM] Player submitted name: Carl
[5/15/2013 2:17:50 PM] Leaving sequence: slide2, moving forward.
[5/15/2013 2:17:55 PM] Player submitted name: Carl
[5/15/2013 2:17:55 PM] Leaving sequence: slide2b, moving forward.
[5/15/2013 2:18:34 PM] Player submitted name: Carl
[5/15/2013 2:18:34 PM] Leaving sequence: slide2c, moving forward.
[5/15/2013 2:20:09 PM] Player submitted name: Carl
[5/15/2013 2:20:09 PM] Leaving sequence: slide3, moving forward.
[5/15/2013 2:20:13 PM] Player submitted name: Carl
[5/15/2013 2:20:13 PM] Leaving sequence: slide4, moving forward.
```

Fig. 8.2 A snippet of an unstructured log from a simulation-based assessment task

subsections, we will show specific examples of how to convert the sequential data, either structured or unstructured, into a tabular data frame for further analysis.

8.3.3 Parsing Unstructured Process Data

Some real-world sequential data are presented in unstructured format, though most of which are usually the results of inappropriate data engineering. Figure 8.2 shows a snippet of an unstructured log file from a simulation-based task. The data are stored in a plain text file named *unstructured_example_log.txt*. Each line of the file has a timestamp and some description about what happened at that time point.

Python provides a set of functionalities to read in and manipulate the fields for such a plain text file. We will show how to parse and rearrange the fields of the sequential data into a tabular data frame. Figure 8.3 shows how to load the needed packages and read in the text file as a list of strings. The input cell In[1] shows how to load the needed packages. The input cell In[2] shows how to open the text file

Step 1. Loading the needed packages

```
In [1]: import pandas as pd
        import numpy as np
        from datetime import datetime
```

Step 2. Open the unstructured data file and read it into a python list

```
In [2]: with open('unstructured_example_log.txt') as f:
            txt = f.readlines()

In [3]: txt

Out[3]: ['[5/15/2013 2:17:26 PM] Session Start\n',
         '[5/15/2013 2:17:26 PM] Leaving sequence: loadXML, moving forward.\n',
         '[5/15/2013 2:17:30 PM] Player submitted name: Carl\n',
         '[5/15/2013 2:17:30 PM] Leaving sequence: InputNameScreen, moving forward.\n',
         '[5/15/2013 2:17:31 PM] Player submitted name: Carl\n',
         '[5/15/2013 2:17:31 PM] Leaving sequence: startScreen, moving forward.\n',
         '[5/15/2013 2:17:50 PM] Player submitted name: Carl\n',
         '[5/15/2013 2:17:50 PM] Leaving sequence: slide2, moving forward.\n',
         '[5/15/2013 2:17:55 PM] Player submitted name: Carl\n',
         '[5/15/2013 2:17:55 PM] Leaving sequence: slide2b, moving forward.\n',
         '[5/15/2013 2:18:34 PM] Player submitted name: Carl\n',
         '[5/15/2013 2:18:34 PM] Leaving sequence: slide2c, moving forward.\n',
         '[5/15/2013 2:20:09 PM] Player submitted name: Carl\n',
         '[5/15/2013 2:20:09 PM] Leaving sequence: slide3, moving forward.\n',
         '[5/15/2013 2:20:13 PM] Player submitted name: Carl\n',
         '[5/15/2013 2:20:13 PM] Leaving sequence: slide4, moving forward.\n']

In [4]: txt[0][0]

Out[4]: '['
```

Fig. 8.3 Loading the needed package and read in a text file as a list of strings

and read the lines in the file as a list of strings in Python and store them into a variable, *txt*.[3] The output cell Out[3] prints out the list *txt* and the output cell Out[4] shows how to reference the first character of the first element in the list variable *txt*. In this way, one can reference or index any character in the text file and can, therefore, manipulate the data at any desired granularity.

Python provides powerful functionalities for processing strings. In the output cell Out[3] of Fig. 8.3, each element of the list ends with a line break character \n, which is usually irrelevant to further analysis. Figure 8.4 shows how to remove them and get a clean list of strings.

Readers may have noticed that the time in the data file is in a string format that does not allow numerical computation directly. Python has a DateTime object to store date and time information and allows numerical operations. Figure 8.5 shows the code snippet for extracting, cleaning, and converting the date and time strings to Python DateTime objects, and then calculates the time difference between two entries in the unit of second.

[3] Python list is specific variable type in the form of [a, b, c, ...].

Step 3. Clean each line to strip off the \n

```
In [5]: txt = [t.strip() for t in txt]

In [6]: txt

Out[6]: ['[5/15/2013 2:17:26 PM] Session Start',
         '[5/15/2013 2:17:26 PM] Leaving sequence: loadXML, moving forward.',
         '[5/15/2013 2:17:30 PM] Player submitted name: Carl',
         '[5/15/2013 2:17:30 PM] Leaving sequence: InputNameScreen, moving forward.',
         '[5/15/2013 2:17:31 PM] Player submitted name: Carl',
         '[5/15/2013 2:17:31 PM] Leaving sequence: startScreen, moving forward.',
         '[5/15/2013 2:17:50 PM] Player submitted name: Carl',
         '[5/15/2013 2:17:50 PM] Leaving sequence: slide2, moving forward.',
         '[5/15/2013 2:17:55 PM] Player submitted name: Carl',
         '[5/15/2013 2:17:55 PM] Leaving sequence: slide2b, moving forward.',
         '[5/15/2013 2:18:34 PM] Player submitted name: Carl',
         '[5/15/2013 2:18:34 PM] Leaving sequence: slide2c, moving forward.',
         '[5/15/2013 2:20:09 PM] Player submitted name: Carl',
         '[5/15/2013 2:20:09 PM] Leaving sequence: slide3, moving forward.',
         '[5/15/2013 2:20:13 PM] Player submitted name: Carl',
         '[5/15/2013 2:20:13 PM] Leaving sequence: slide4, moving forward.']
```

Fig. 8.4 Python code example of cleaning strings in Python

Step 4. Sepate the time stamp and convert it to standard Python datetime object

```
In [7]: s0 = txt[0].split(']')[0].strip('[')
        s10 = txt[10].split(']')[0].strip('[')

In [8]: s0

Out[8]: '5/15/2013 2:17:26 PM'

In [9]: s10

Out[9]: '5/15/2013 2:18:34 PM'

In [10]: dtfmt ='%m/%d/%Y %I:%M:%S %p'   # %H -> 24 hours, %I-> 12 hours
         t0 = datetime.strptime(s0, dtfmt)
         t10 = datetime.strptime(s10, dtfmt)

In [11]: t0

Out[11]: datetime.datetime(2013, 5, 15, 14, 17, 26)

In [12]: t10

Out[12]: datetime.datetime(2013, 5, 15, 14, 18, 34)

In [13]: (t10-t0).seconds

Out[13]: 68
```

Fig. 8.5 Code snippet for converting date time string to Python DateTime object

Next, in Fig. 8.6, we show the codes that can restructure the data into a Pandas data frame. In the input cell In[14], we define a function where each line of the text file is divided into three chunks and stored into three lists, session_time, event_name, and event_attribute. The lists are eventually converted into a Pandas data frame and the corresponding output cell Out[15] shows the resulting data frame.

Step 5. Restructure the information into a data frame

In [14]:
```
# define a function to combine all the above steps and turn the results into a pandas
Data Frame

def generate_data_frame(txt):
    session_time = []
    event_name = []
    event_attribute = []
    dtfmt ='%m/%d/%Y %I:%M:%S %p'
    for line in txt:
        s1=line.split(']')[0].strip('[')
        dt = datetime.strptime(s1, dtfmt)
        session_time.append(dt)
        s= line.split(']')[1].strip().split(':')
        event_name.append(s[0])
        if len(s) == 2:
            event_attribute.append(s[1].lstrip())
        else:
            event_attribute.append(np.nan)
    df = pd.DataFrame([session_time,event_name,event_attribute]).T
    df.columns=['session_time','event_name', 'event_attribute']
    return df
```

In [15]:
```
generate_data_frame(txt)
```

Out[15]:

	session_time	event_name	event_attribute
0	2013-05-15 14:17:26	Session Start	NaN
1	2013-05-15 14:17:26	Leaving sequence	loadXML, moving forward.
2	2013-05-15 14:17:30	Player submitted name	Carl
3	2013-05-15 14:17:30	Leaving sequence	InputNameScreen, moving forward.
4	2013-05-15 14:17:31	Player submitted name	Carl
5	2013-05-15 14:17:31	Leaving sequence	startScreen, moving forward.
6	2013-05-15 14:17:50	Player submitted name	Carl
7	2013-05-15 14:17:50	Leaving sequence	slide2, moving forward.
8	2013-05-15 14:17:55	Player submitted name	Carl
9	2013-05-15 14:17:55	Leaving sequence	slide2b, moving forward.
10	2013-05-15 14:18:34	Player submitted name	Carl
11	2013-05-15 14:18:34	Leaving sequence	slide2c, moving forward.
12	2013-05-15 14:20:09	Player submitted name	Carl
13	2013-05-15 14:20:09	Leaving sequence	slide3, moving forward.
14	2013-05-15 14:20:13	Player submitted name	Carl
15	2013-05-15 14:20:13	Leaving sequence	slide4, moving forward.

Fig. 8.6 Codes snippet for converting unstructured data into a Pandas data frame

At this point, one can apply additional operations on the in-memory data frame to extract the desired evidence. Note that the codes snippets here are motivated by logical clarity rather than execution efficiency. They are not necessarily the most efficient way to accomplish the goals. We encourage readers to explore different ways to complete the above exercise.

```
{"data": [{"session_time": "2013-05-15 14:17:26", "event_name": "Session Start", "event_attribute":
"NaN"}, {"session_time": "2013-05-15 14:17:26", "event_name": "Leaving sequence", "event_attribute":
"loadXML, moving forward."}, {"session_time": "2013-05-15 14:17:30", "event_name": "Player submitted
name", "event_attribute": "Carl"}, {"session_time": "2013-05-15 14:17:30", "event_name": "Leaving
sequence", "event_attribute": "InputNameScreen, moving forward."}, {"session_time": "2013-05-15
14:17:31", "event_name": "Player submitted name", "event_attribute": "Carl"}, {"session_time":
"2013-05-15 14:17:31", "event_name": "Leaving sequence", "event_attribute": "startScreen, moving
forward."}, {"session_time": "2013-05-15 14:17:50", "event_name": "Player submitted name",
"event_attribute": "Carl"}, {"session_time": "2013-05-15 14:17:50", "event_name": "Leaving sequence",
"event_attribute": "slide2, moving forward."}, {"session_time": "2013-05-15 14:17:55", "event_name":
"Player submitted name", "event_attribute": "Carl"}, {"session_time": "2013-05-15 14:17:55",
"event_name": "Leaving sequence", "event_attribute": "slide2b, moving forward."}, {"session_time":
"2013-05-15 14:18:34", "event_name": "Player submitted name", "event_attribute": "Carl"},
{"session_time": "2013-05-15 14:18:34", "event_name": "Leaving sequence", "event_attribute": "slide2c,
moving forward."}, {"session_time": "2013-05-15 14:20:09", "event_name": "Player submitted name",
"event_attribute": "Carl"}, {"session_time": "2013-05-15 14:20:09", "event_name": "Leaving sequence",
"event_attribute": "slide3, moving forward."}, {"session_time": "2013-05-15 14:20:13", "event_name":
"Player submitted name", "event_attribute": "Carl"}, {"session_time": "2013-05-15 14:20:13",
"event_name": "Leaving sequence", "event_attribute": "slide4, moving forward."}]}
```


Fig. 8.7 A snippet of a structured log in JSON format from a simulation-based assessment task

8.3.4 Parsing Structured Process Data

The data from most well-designed DBAs are structured, though not necessarily well-structured. JSON and XML are two most popular formats used to store such data, and the specifications are instantiated as JSON or XML schema. The schema can be used for compliance check against the incoming data, providing basic data quality assurance to ensure data integrity. There are several Python packages to assist the parsing of structured data in JSON and XML format. In this subsection, we use an example to show how to parse and restructure a structured data file in JSON format into a Pandas data frame. The same example data used in the previous subsection have been cast into a structured JSON file named *structured_example_log.json,* as shown in Fig. 8.7

As with parsing unstructured data, we must first load some needed packages. In Python, there is a package called "json" that can read in a JSON file as a Python dictionary.[4] The corresponding code snippet is shown in Fig. 8.7. The input cell In[1] loads the needed packages. The input cell In[2] opens the JSON file and reads it in as a python dictionary named *txt*. The output cell Out[3] prints out the *txt* dictionary. Pandas provides a convenient way for converting a list of dictionaries into a data frame, as illustrated by the codes shown in Figs. 8.8 and 8.9. These are examples based on the file in JSON format; similar methods exist for processing XML files. We include the corresponding Jupyter Notebook in the GitHub code repository.

[4]Python dictionary is a specific variable type in the form of key-value pairs as {"key1": "value1", "key2": "value2", . . . }.

Step 1. Loading the needed packages

```
In [1]: import pandas as pd
        import json
        from datetime import datetime
```

Step 2. Open the structured data file and read it into a Python dictionary

```
In [2]: with open('structured_example_log.json') as f:
            txt = json.load(f)

In [3]: txt

Out[3]: {'data': [{'session_time': '2013-05-15 14:17:26',
           'event_name': 'Session Start',
           'event_attribute': 'NaN'},
          {'session_time': '2013-05-15 14:17:26',
           'event_name': 'Leaving sequence',
           'event_attribute': 'loadXML, moving forward.'},
          {'session_time': '2013-05-15 14:17:30',
           'event_name': 'Player submitted name',
           'event_attribute': 'Carl'},
          {'session_time': '2013-05-15 14:17:30',
           'event_name': 'Leaving sequence',
           'event_attribute': 'InputNameScreen, moving forward.'},
          {'session_time': '2013-05-15 14:17:31',
           'event_name': 'Player submitted name',
           'event_attribute': 'Carl'},
          {'session_time': '2013-05-15 14:17:31',
           'event_name': 'Leaving sequence',
           'event_attribute': 'startScreen, moving forward.'},
          {'session_time': '2013-05-15 14:17:50',
           'event_name': 'Player submitted name',
           'event_attribute': 'Carl'},
          {'session_time': '2013-05-15 14:17:50',
           'event_name': 'Leaving sequence',
           'event_attribute': 'slide2, moving forward.'},
          {'session_time': '2013-05-15 14:17:55',
           'event_name': 'Player submitted name',
           'event_attribute': 'Carl'},
          {'session_time': '2013-05-15 14:17:55',
           'event_name': 'Leaving sequence',
           'event_attribute': 'slide2b, moving forward.'},
          {'session_time': '2013-05-15 14:18:34',
           'event_name': 'Player submitted name',
           'event_attribute': 'Carl'},
```

Fig. 8.8 Code snippet for loading the JSON package and read in the JSON file as a Python dictionary

8.3.5 Microbatch and Stream Processing via Generator Function

So far, we have introduced some basic techniques for restructuring raw sequential data into a manageable tabular data frame for further processing (often in batch).

Step 3. Convert a list of dictionaries to data frame

```
In [4]: pd.DataFrame(txt.get('data'))
```

Out[4]:

	session_time	event_name	event_attribute
0	2013-05-15 14:17:26	Session Start	NaN
1	2013-05-15 14:17:26	Leaving sequence	loadXML, moving forward.
2	2013-05-15 14:17:30	Player submitted name	Carl
3	2013-05-15 14:17:30	Leaving sequence	InputNameScreen, moving forward.
4	2013-05-15 14:17:31	Player submitted name	Carl
5	2013-05-15 14:17:31	Leaving sequence	startScreen, moving forward.
6	2013-05-15 14:17:50	Player submitted name	Carl
7	2013-05-15 14:17:50	Leaving sequence	slide2, moving forward.
8	2013-05-15 14:17:55	Player submitted name	Carl
9	2013-05-15 14:17:55	Leaving sequence	slide2b, moving forward.
10	2013-05-15 14:18:34	Player submitted name	Carl
11	2013-05-15 14:18:34	Leaving sequence	slide2c, moving forward.
12	2013-05-15 14:20:09	Player submitted name	Carl
13	2013-05-15 14:20:09	Leaving sequence	slide3, moving forward.
14	2013-05-15 14:20:13	Player submitted name	Carl
15	2013-05-15 14:20:13	Leaving sequence	slide4, moving forward.

Fig. 8.9 Python code to convert a list of dictionaries to a data frame

The methods generally work well when the data size is not very large so that they can be read into the memory of a computer and all the data are available before performing the processing. However, if the data size if very large, more than the memory of a typical modern computer, or if new data keep being added continuously, we need to consider different processing strategies, such as microbatch and stream processing. In this subsection, we introduce how to handle very large and growing log files using the generator function in Python through microbatch processing and stream processing respectively.

To be specific, let's consider a hypothetical log file similar to the unstructured log file in Sect. 8.3.3, but has one trillion lines instead of sixteen. If one runs the scripts as shown in Fig. 8.2, it is likely to get into issues after executing the In[2] cell as the function `f.readlines()` attempts to read all lines in this one trillion-line file into the computer memory. A more favorable strategy may be processing the data in chunks of smaller size when the processing function is called and then release the corresponding computer memory after processing each chunk. Such a strategy is referred to as lazy evaluation in programming language theory (Reynolds, 2009). In Python, generator functions provide an elegant way to accomplish lazy evaluation. There are many nice tutorials about the generator function in Python (e.g., Beazley, 2018); we refer readers to them for further details. In Fig. 8.10, we show some code examples of using generator functions to parse and restructure the unstructured log

Step 1. Loading the needed packages

In [1]:
```
import pandas as pd
import numpy as np
from datetime import datetime
from itertools import islice
import sys
```

Step 2. Define functions

In [2]:
```
# define generator function to read in files in chuncks
def read_chunks(file_obj,chunk_size):
    while True:
        lines = list(islice(file_obj, chunk_size))
        if lines:
            yield lines
        else:
            print('end of file')
            break
```

Step 3. Reload the previously defined function

In [3]:
```
# this function is the same as the previous one but we reload it here
def generate_data_frame(txt):
    session_time = []
    event_name = []
    event_attribute = []
    dtfmt ='%m/%d/%Y %I:%M:%S %p'
    for line in txt:
        s1=line.split(']')[0].strip('[')
        dt = datetime.strptime(s1, dtfmt)
        session_time.append(dt)
        s= line.split(']')[1].strip().split(':')
        event_name.append(s[0])
        if len(s) == 2:
            event_attribute.append(s[1].lstrip())
        else:
            event_attribute.append(np.nan)
    df = pd.DataFrame([session_time,event_name,event_attribute]).T
    df.columns=['session_time','event_name', 'event_attribute']
    return df
```

Fig. 8.10 Code snippet of the stream processing pipeline using generator functions for the unstructured data file

used in Sect. 8.3.3 to give readers a sense of a microbatch processing pipeline for very large data files.

The input cell In[2] formally defines a generator function `df` that is essentially a set of procedures to read in a chunk of the unstructured data file and convert it to a data frame. The input cell In[7] uses the "`next`" function to trigger the `df` to take action. One can repeatedly apply the `next` function or use a loop to exhaust all the lines in the log file without leading to a memory overflow error, which is desirable for designing a scalable processing pipeline. It is worth noting that the generator-based stream processing is slightly slower compared to a batch processing for smaller data file where the computer memory is not a concern. In practice, readers need to make their own decisions for choosing which processing pipeline to use, based on the data size, intended use, and project priorities.

The above method for continuously processing a very large file by dividing it into smaller batches is known as microbatch processing. It is something in between of batch processing and stream processing. Typical stream processing, however,

Step 4. Open files and read data into a generator object

```
In [4]: # open the file
        f = open('../data/unstructured_example_log.txt','r')

In [5]: # read file into chunks of 4 lines and create a generator object
        chunk_generator = read_chunks(f,4)

In [6]: # check the memory usage of the generator object in bytes
        sys.getsizeof(chunk_generator)

Out[6]: 128

In [7]: # get the first chunk using next()
        next(chunk_generator)

Out[7]: ['[5/15/2013 2:17:26 PM] Session Start\n',
         '[5/15/2013 2:17:26 PM] Leaving sequence: loadXML, moving forward.\n',
         '[5/15/2013 2:17:30 PM] Player submitted name: Carl\n',
         '[5/15/2013 2:17:30 PM] Leaving sequence: InputNameScreen, moving forward.\n']

In [8]: # create a new generator function called df
        df = (generate_data_frame(txt) for txt in chunk_generator)

In [9]: # check the memory usage of the generator object in bytes
        sys.getsizeof(df)

Out[9]: 128

In [10]: # get the first object in dfnext(df)
         next(df)

Out[10]:
```

	session_time	event_name	event_attribute
0	2013-05-15 14:17:31	Player submitted name	Carl
1	2013-05-15 14:17:31	Leaving sequence	startScreen, moving forward.
2	2013-05-15 14:17:50	Player submitted name	Carl
3	2013-05-15 14:17:50	Leaving sequence	slide2, moving forward.

```
In [11]: f.close()
```

Fig. 8.10 (continued)

usually deals with event data that keep coming, leading to files with potentially "infinite" size. For example, in an online shopping website, such as Amazon.com, customers from all over the world make various requests (e.g., put things into their shopping carts, make payments, and so on) from time to time without a clear ending point. Each request requires a near real time processing that leads to actions to address the customer's needs. In practice, there are software infrastructures specially designed for stream data and application programming interfaces (APIs) are provided to allow users to interact with the data stream. An example of such implementations is the twitter data stream API (readers can get more details from the website (https://help.twitter.com/en/rules-and-policies/twitter-api)). In the following, as a simplified example of stream processing, we show how to process a growing log file. First, we create a growing log file (stream_data.txt) by adding a new line to it every two seconds. The python codes and part of the log file are shown in Fig. 8.11.

```python
# ---- stream_data_generation.py ---
# generating a growing data file

import time

if __name__=="__main__":
    line_number = 1
    while True:
        file_stream = open("stream_data.txt", "a")
        contents = '-- This is Line: '+str(line_number)+' -- \n'
        file_stream.write(contents)
        file_stream.close()
        time.sleep(2)
        line_number = line_number +1
        print(line_number)
        if line_number == 3600:
            break
        continue
```

stream_data.txt

```
-- This is Line: 1 --
-- This is Line: 2 --
-- This is Line: 3 --
-- This is Line: 4 --
-- This is Line: 5 --
-- This is Line: 6 --
-- This is Line: 7 --
-- This is Line: 8 --
-- This is Line: 9 --
-- This is Line: 10 --
-- This is Line: 11 --
-- This is Line: 12 --
-- This is Line: 13 --
-- This is Line: 14 --
-- This is Line: 15 --
-- This is Line: 16 --
-- This is Line: 17 --
-- This is Line: 18 --
```

Fig. 8.11 Python codes (left) that add a new line of text to a log file every two seconds and the resulting log file (right)

Next, we create a generator function to read the growing log file, either from the beginning or from the current ending point, and then iterate through the (potentially infinite) lines one by one. One can perform any filtering or processing on each line while iterating through the lines. The codes are shown in Fig. 8.12.

The above example is highly simplified, but it captures the essential feature of stream processing via generator functions in Python. In real-world applications, different software infrastructures are used for different types of data and processing. Users can directly use the corresponding APIs instead of programming from scratch using generators. For example, Apache Spark (https://spark.apache.org/) is designed for dealing with microbatch processing while Apache Kafka (https://kafka.apache.org/) and Flink (https://flink.apache.org/) are designed for handling stream processing. They all provide high-level APIs to massage the data stream. We recommend readers to consult the corresponding documentations for more details.

8.4 Methodological Strategies for Making Sense of Process Data

Most familiar psychometric methodologies, such as IRT and classical test theory (CTT), have been developed for simpler and regular assessment data that can be represented as a response/score table (Chap. 6). As argued in Chap. 3, however, often these methods cannot be applied, in their basic forms, to the more complex process data arising from DBAs. As such, new methodological strategies are needed to guide the sensible use of process data to support assessments. We envision the uses of process data from DBAs can be put into three broad categories, then discuss corresponding methodological strategies for each of them in the following.

Step 1. Loading the needed packages

```
In [1]: import time
```

Step 2. Define a generator function to get the latest log entries

```
In [2]: # Example function modified from David Beazley (https://www.dabeaz.com/generators/Generators.pdf)

def follow(thefile,start='head'):
    if start == 'head':
        thefile.seek(0,0) # start from the beginning of the file
    if start == 'tail':
        thefile.seek(0,2) # start from the current ending point of the file
    while True:
        line = thefile.readline()
        if not line:
            time.sleep(0.1) # Sleep briefly
            continue
        yield line
```

Step 3. Working on the lines of the growing file

```
In [9]: # open the log file
logfile = open('../data/stream_data.txt')

In [10]: # follow all lines in the growing file
line_all = follow(logfile,start='head')

In [5]: # get out the lines sequentially
next(line_all)

Out[5]: '-- This is Line: 1 -- \n'

In [6]: # follow the new lines starting from the execution of the function
line_latest = follow(logfile,start='tail')

In [7]: next(line_latest)

Out[7]: '-- This is Line: 8 -- \n'

In [8]: # close the log file
logfile.close()
```

Fig. 8.12 A Jupyter Notebook showing codes to handle a growing log file

8.4.1 Process Data as Augmenting Standard Assessments

DBAs can be at varying levels of complexity. The simplest kind of DBA is an assessment that is very similar to a familiar paper-based assessment but delivered in a digital environment. Usually, the intended use and method of scoring are the same as they would be in a nondigital form, based on designated features of final products such as correctness of multiple-choice responses or holistic or analytic scores of constructed verbal responses. The digital environment, however, makes it possible to capture more detailed aspects of the performances, such as timestamped individual keystrokes, mouse clicks to maneuver among parts of the stimulus materials, changed answer choices, moves from task to task, and deletions, additions, and movements of text elements in verbal responses. This subsection

addresses the use of process data to improve or modestly extend traditional forms of testing and construct definition—that is, the enacted construct is operationally defined in terms of traditional aspects of performance, such as correct selections or ratings of final written responses. Even when only final products and scoring methods are required for operational use; however, the augmented data hold much information for other purposes, notably examining data quality, improving tasks, and investigating validity (Ercikan & Pellegrino, 2017; Zumbo & Hubley, 2017). Seeing exactly how test takers have interacted with the interface adds insight into thinking processes for all students, information that was formerly available on a smaller scale through think-aloud studies, human observers in real-time, or time-consuming analyses of video captures.

The most straightforward strategy in such situations is to turn the complex process data into a score table-like representation with rows representing students and columns representing feature variables characterizing aspects of the process (e.g., patterns), which we referred to as the feature table hereafter. The general approaches to identify patterns to form feature representations is often known as analytics, which is defined as *the discovery, interpretation, and communication of meaningful patterns in data; and the process of applying those patterns towards effective decision making* (Wikipedia contributors, 2019). The feature table resembles the score table that is familiar to most psychometricians except that its elements cannot be simply interpreted as scores in psychometric sense.[5] The score table generally captures the information regarding the outcomes of the person-item interaction, while the feature table summarizes additional information about the process of the person-item interaction (e.g., response time). Generally speaking, the techniques for dealing with the score tables used in traditional psychometrics can also be applied to the feature table, which could lead to new item parameters in addition to the well-known ones, such as item difficulty and differentiability, based on the score table (e.g., regression parameters in an IRT model in which response time provides auxiliary information about test-takers' propensity to response correctly; see Roskam, 1997).

This is not a new strategy, and psychometricians have been analyzing constructed response items using a similar strategy for a long time, though we may not think of it from this perspective. The text responses to a constructed response item can be viewed as a response process, e.g., a sequence of timestamped words, even though the timestamps are usually not explicitly present in paper-pencil tests. In this case, human raters with a scoring rubric will map the response sequence into scores. The response process data from DBAs are very similar to the word sequences from constructed responses except for the explicit timestamps. Computers and algorithms will replace human raters and scoring rubrics to map the response sequences to feature representations. In this regard, the research focus will be on the identification of appropriate algorithms or stochastic models that can map a sequence to a feature or map several subsequences to multiple features.

[5] In some cases, the features can be considered as scores as well.

Identifying appropriate algorithms to turn response process data into meaningful features is more an art than science; some are discussed in other chapters of this volume. For example, suppose we have a DBA with 20 selected response items, and we capture test takers' response process information in addition to the final responses. Based on the response process data, what features can we develop to characterize the process? There could be many from different perspectives. For example, the response time to each item, the number of visits to each item, the order of visiting items, and the transition from item to item, could all be process-based features, which may contain information relevant to the test taker's proficiency or the item properties. This additional information can help in detecting differences across demographic groups, improving estimates of test takers' proficiency, or assisting item calibration by identifying aberrant responses to remove from the calibration process. Many more complex algorithms and stochastic models have been explored in the literature to accomplish this goal, such as the social network analysis described in Chap. 13 and the edit distance approach (Hao et al., 2015). This methodological strategy is most suitable for DBAs consisting of relatively short response process data, though the analytic methods can also be applied to interactive simulation-based tasks as auxiliary data, even when process data are used further as evidence for inferences about test takers' capabilities, as discussed in the following sections.

8.4.2 Process Data as Evidence in Psychometric Models

A second methodological strategy is directly relating the evidence from process data to constructs operationalized as latent variables in a psychometric measurement model, such as the cases discussed in Chaps. 2 and 4. The model itself, at its highest layer, could be a familiar one such as IRT, cognitive diagnosis, or latent class analysis if the data suit their assumptions; it could be an extension of such models, such as Bayes nets (static or dynamic; see Chap. 3), hidden Markov processes, or learning models as used in simulation-based assessments or intelligent tutoring systems (Koedinger et al., 2012; VanLehn, 2008), or multivariate models used to synthesize evidence in multimodal assessments (Khan, 2017). As discussed in Chap. 2, the process of moving from low-level process data to evidence entered into a latent-variable psychometric model may be carried out in multiple stages (also see Kerr et al., 2016; Khan, 2017; Mislevy, 2018).

The distinguishing feature of this strategy, however, is that the process data are used as evidence for the latent variables, and as such directly affect the *meaning* of those variables. That is, their operationalized meanings can extend beyond the kind of evidence in traditional forms of data, and directly express aspects of performance not directly captured in traditional data, such as strategy choices, efficiency, and responses to changing situations. Process data can in these ways enrich, or in some cases even make possible, the assessment of constructs that lie beyond the scope of traditional assessment methods.

This strategy is aligned with ECD and is better suited for VPAs such as simulation-based and game-based assessments. Examples of its application with particular computational-psychometric methods appear in Chaps. 2 and 4. As mentioned, this modeling strategy can be developed in conjunction with the more exploratory and auxiliary computational-psychometric analyses discussed in the previous section (DiCerbo et al., 2015).

8.4.3 Process Data as Evidence in Cognitive Models

The third methodological strategy is directly mapping the response process or selected features of it to an underlying cognitive model. Methods under this strategy directly map the events or event sequences in the response process to parameters of a cognitive processing model rather than to either intermediate feature representations or latent variables representing higher-level constructs. One example of this strategy is fitting production-rule models from information-processing cognitive psychology to a test taker's sequence of actions. Singley (1995) illustrates the idea with a model-tracing algorithm to infer students' goals and strategies from their step-by-step solutions of open-ended algebra problems. Another example is fitting a Markov decision process model to a test taker's sequence of actions, as functions of the test taker's goals, knowledge, and efficiency of actions. LaMar (2018) illustrates this model in the context of a game-based strategy assessment. Chap. 11 offers further discussion of this approach.

This strategy too can be combined with methods from the previous two subsections. The exploratory analyses discussed in Sect. 8.4.1 are useful whenever complex data are encountered, to better understand the nature of performance in the complex environment (and, often, to cycle back among design and theorizing). When models such as the ones mentioned in this section are used to model performance *within tasks*, a psychometric model can be then built with higher-level constructs presumed to influence performance *across tasks.* Hierarchically, the within-task latent variables are models as probabilistically dependent on the across-task latent variables. This is a style of implementing the strategy of Sect. 8.4.2.

8.5 Summary

Advances in psychology have provided us with a better understanding of the nature of human capabilities, and how we acquire them and develop them in a complex adaptive sociocognitive system. Advances in technology enable us to design and implement digital environments in which students can interact with realistic environments and other students, in which we can elicit rich performances that should be able to provide evidence about those capabilities. What stands between our targeted inferences and students' rich performances? Design, data,

and analyses. This chapter brings *data* to the foreground: how, in concert with psychology, technology, design, and computational-psychometric modeling, we need to think about data, if we are to realize the potential for a new generation of assessment awaits us.

Though we summarized three broad strategies towards modeling process data from DBAs, readers should be aware that the process data from a real-world DBA often have some unique properties that may add additional complications to applying these strategies in practice. Four typical complications concerned with modeling real-world DBA data are as follows. First, response processes are usually subject to the specific task design, which can limit the generalizability of the algorithms/models across tasks. Second, an extended performance can be the result of multiple strategies in different parts of the performance, requiring different regimes that correspond to different stationary conditions for temporal modeling. Third, the mathematical space spanned by the response sequence is usually very big and sparse—even sometimes for the augmented files for multiple-choice tests, when timing, ordering, and revisions are captured—which makes it challenging even to identify a suitable statistical distribution to model randomness in the sequences. Fourth, typical response sequences to DBAs that are similar to traditional task forms are often short, which makes it challenging to estimate the parameters of stochastic models. As such, we suggest that readers who work with real data have a thorough understanding of the data at hand before delving into applying different modeling strategies.

References

Almond, R. G., Steinberg, L. S., & Mislevy, R. J. (2002). Enhancing the design and delivery of assessment systems: A four-process architecture. *The Journal of Technology, Learning and Assessment, 1*(5).

Baker, R. S., Corbett, A. T., Koedinger, K. R., & Wagner, A. Z. (2004). Off-task behavior in the cognitive tutor classroom: When students game the system. *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 383–390). New York, NY: Association for Computing Machinery.

Beazley, D. (2018). Generator tricks for systems programmers, Retrieved September 2nd, 2019, from https://www.dabeaz.com/generators/Generators.pdf

Deane, P. (2014). Using writing process and product features to assess writing quality and explore how those features relate to other literacy tasks. *ETS Research Report Series, 2014*(1), 1–23.

DiCerbo, K., Bertling, M., Stephenson, S., Jia, Y., Mislevy, R. J., Bauer, M., & Jackson, T. (2015). The role of exploratory data analysis in the development of game-based assessments. In C. S. Loh, Y. Sheng, & D. Ifenthaler (Eds.), *Serious games analytics: Methodologies for performance measurement, assessment, and improvement* (pp. 319–342). Springer.

Ercikan, K., & Pellegrino, J. W. (2017). *Validation of score meaning for the next generation of assessments: The use of response processes*. Taylor & Francis.

Halverson, R., & Owen, V. E. (2014). Game-based assessment: an integrated model for capturing evidence of learning in play. *International Journal of Learning Technology, 9*(2), 111–138.

Hao, J., & Mislevy, R. J. (2018). The evidence trace file: A data structure for virtual performance assessments informed by data analytics and evidence-centered design. *ETS Research Report Series, 2018*(1), 1–16.

Hao, J., Smith III, L.L., Mislevy, R.J., & von Davier, A.A. (2014). Systems and methods for designing, parsing and mining of game log files. U.S. patent application # 14/527,591.

Hao, J., Shu, Z., & von Davier, A. (2015). Analyzing process data from game/scenario-based tasks: An edit distance approach. *Journal of Educational Data Mining, 7*(1), 33–50.

Hao, J., Smith, L., Mislevy, R., von Davier, A., & Bauer, M. (2016). Taming log files from game/simulation-based assessments: Data models and data analysis tools. *ETS Research Report Series, 2016*(1), 1–18.

IMS Global Learning Consortium. (2015). Caliper analytics. Retrieved from the website of IMS Global Learning Consortium, http://www.imsglobal.org/activity/caliper

LaMar, M. M. (2018). Markov decision process measurement model. *Psychometrika, 83*(1), 67–88.

Kerr, D., Andrews, J. J., & Mislevy, R. J. (2016). The in-task assessment framework for behavioral data. In A. Rupp & J. Leighton (Eds.), *Handbook of cognition and assessment* (pp. 472–507). Wiley-Blackwell.

Kevan, J. M., & Ryan, P. R. (2016). Experience API: Flexible, decentralized and activity-centric data collection. *Technology, Knowledge and Learning, 21*(1), 143–149.

Khan, S. M. (2017). Multimodal behavioral analytics in intelligent learning and assessment systems. In A. A. von Davier, M. Zhu, & P. C. Kyllonen (Eds.), *Innovative assessment of collaboration* (pp. 173–184). Springer International.

Kleppmann, M. (2017). *Designing data-intensive applications: The big ideas behind reliable, scalable, and maintainable systems*. O'Reilly Media, Inc..

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B. E., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J., Grout, J., Corlay, S., Ivanov, P., Avila, D., Abdalla, S., Willing, C., & Jupyter Development Team. (2016). Jupyter notebooks – A publishing format for reproducible computational workflows. In F. Loizides & B. Schmidt (Eds.), *Positioning and power in academic publishing: Players, agents and agendas* (pp. 87–90). IOS Press.

Koedinger, K. R., Corbett, A. T., & Perfetti, C. (2012). The Knowledge-Learning-Instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive Science, 36*(5), 757–798.

Mislevy, R. J. (2018). Sociocognitive foundations of educational measurement. New York/London: Routledge.

Mislevy, R. J., Almond, R. G., & Lukas, J. F. (2003). A brief introduction to evidence-centered design. *ETS Research Report Series, 2003*(1), i–29.

Mislevy, R., Oranje, A., Bauer, M. I., von Davier, A. A., Hao, J., Corrigan, S., Hoffman, E., DiCerbo, K., & John, M. (2014). *Psychometric considerations in game-based assessments*. CreateSpace.

Reynolds, J. C. (2009). *Theories of programming languages*. Cambridge University Press.

Roberts, J. D., Chung, G. K. W. K., & Parks, C. B. (2016). Supporting children's progress through the PBS KIDS learning analytics platform. *Journal of Children and Media, 10*, 257–266.

Robinson, D. (2017, Sept. 6), The incredible growth of Python. Retrieved from https://stackoverflow.blog/2017/09/06/incredible-growth-python/

Roskam, E. E. (1997). Models for speed and time-limit tests. In W. J. van der Linden & R. K. Hambleton (Eds.), *Handbook of modern item response theory* (pp. 187–208). Springer.

Singley, M. K. (1995). Promoting transfer through model tracing. In A. McKeough, J. L. Lupart, & A. Marini (Eds.), *Teaching for transfer: Fostering generalization in learning* (pp. 69–92). Erlbaum.

VanLehn, K. (2008). Intelligent tutoring systems for continuous, embedded assessment. In C. A. Dwyer (Ed.), *The future of assessment: Shaping teaching and learning* (pp. 113–138). Erlbaum.

Wikipedia Contributors. (2018, December 29). Data science. In *Wikipedia, The Free Encyclopedia.* Retrieved 18:00, January 6, 2019, Retrieved from https://en.wikipedia.org/w/index.php?title=Data_science&oldid=875831036

Wikipedia Contributors. (2019a, March 18). Analytics. In *Wikipedia, The Free Encyclopedia.* Retrieved 14:22, April 11, 2019, from https://en.wikipedia.org/w/index.php?title=Analytics&oldid=888300383

Wikipedia Contributors. (2019b, October 29). Data model. In *Wikipedia, The Free Encyclopedia.* Retrieved 20:33, November 7, 2019, from https://en.wikipedia.org/w/index.php?title=Data_model&oldid=923547472

Zumbo, B., & Hubley, A. (Eds.). (2017). *Understanding and investigating response processes in validation research*. Springer.

Chapter 9
Supervised Machine Learning

Jiangang Hao

Abstract Machine learning refers to a set of methodologies that allow computers to "learn" the relationship among numerical representations of data. In this Chapter, we focus on an important branch of machine learning, supervised machine learning, and introduce three widely used supervised learning methods, the Support Vector Machine, Random forest, and Gradient Boosting Machine. Python codes examples are included to show how to use these methods in practice.

9.1 Introduction

As highlighted in Chap. 6, one of the core missions of psychometrics is to ensure the observed evidence from assessment or learning tasks can support the claims on the targeted constructs in a valid, reliable, fair and comparable way. In the area of educational assessment, traditional psychometrics has given most attention to developing systematic methodologies to accomplish this mission when the evidence can be easily mapped into some simple forms, such as dichomotous or polymotous scores. The regularity and simplicity of the data make statistical inference well suited for modeling the data. However, digitally based learning and assessment tasks generate much more complex data. The complexity of these data makes it difficult (or sometimes impossible) to represent the information in simple scores in order to apply well-established statistical modeling frameworks, such as the familiar Item Response Theory (IRT). So, there is an intrinsic need for additional methodologies to harness the more complex data from digitally based

The R or Python codes can be found at the GitHub repository of this book: https://github.com/jgbrainstorm/computational_psychometrics

J. Hao (✉)
Educational Testing Service, Princeton, NJ, USA
e-mail: jhao@ets.org

© Springer Nature Switzerland AG 2021

A. A. von Davier et al. (eds.), *Computational Psychometrics: New Methodologies for a New Generation of Digital Learning and Assessment*, Methodology of Educational Measurement and Assessment, https://doi.org/10.1007/978-3-030-74394-9_9

tasks. Fortunately, the need for "modeling" complex data emerged much earlier in disciplines than psychometrics, and many methodologies have already been developed. In this and the next two consecutive Chapters, we will introduce a set of such powerful methodologies, machine learning, which was developed since late 1950s in the field of computer science and statistics for dealing with complex data.

Machine learning, a term coined by Auther Samuel (Samuel, 1959), refers to a set of methodologies that allow computers to "learn" the relationship among numerical representations of data without explicit instructions by human experts. Based on the learning goals, machine learning tasks are classified into broad categories such as supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning. The numerical representation of data usually can be categorized into features (independent variables) and labels (dependent variables). If a task is to learn the relationship between the features and labels, it is a supervised learning task. If the goal is to discover concentrations, associations, or correlations in the features themselves, the corresponding task is unsupervised learning (e.g., see Chap. 10). Semi-supervised learning refers to the task of learning the mapping between features and labels when only part of the training data has labels. Reinforcement learning is to learn optimal strategies by means of a goal-oriented exploration of a parameter or state-space to optimize a reward function (Sutton & Barto, 1998).

Numerous methods[1] have been developed in each of these categories, and there are many excellent texts that provide comprehensive coverage of them (e.g., Bishop, 2006; Hastie et al., 2009; Witten et al., 2016). It is worth noting that the boundaries of the broad categories above are not always clear-cut, as many methods can be used in more than one types of machine learning task in practice, depending on the ways one trains the algorithms. The current Chapter focuses on supervised machine learning (or supervised learning), Chap. 10 focuses on unsupervised machine learning, and Chap. 11 focuses on deep learning.

9.2 Supervised Learning

Supervised learning refers to a subset of machine learning algorithms that establish a mapping between features and labels of a dataset. The precondition of using supervised learning methods is that both the features and labels are known. Supervised learning methods can be grouped into two categories based on the nature of the labels: regression for continuous labels, and classification for discrete labels. Two primary questions are frequently raised by researchers with psychometric backgrounds when they are introduced to supervised learning. The first is how supervised learning is different from statistical inference, as they both aim at mapping the relationship between the independent and dependent variables. The

[1] A machine learning method is often referred to as a learner in machine learning literature

second is which, if any, supervised learning methods are consistently superior to the others.

For the first question, there are many excellent discussions from different perspectives (e.g., Hastie et al., 2009). Hao and Ho (2019) recently summarized three different emphases between statistical inference and machine learning: machine learning is more prediction-driven while statistical inference emphasizes both prediction and model parameter estimation; statistical inference emphasizes developing a probabilistic model to characterize the data and then estimating the model parameters based on some (usually well-studied) probability distribution functions, while machine learning emphasizes computation algorithms that can efficiently carry out the inference process by minimizing certain loss functions that do not necessarily have a probabilistic underpinning; and statistical inference usually deals with data with a small number of variables obtained through planned *experiments* or quasi-experimental comparisons while machine learning handles sparse and high-dimensional data with a large number of variables (features), usually obtained through passive and uncontrolled *observations*. Despite these different emphases, the distinctions are not always black and white. Increasingly, machine learning makes use of many techniques from statistical modeling.

The second question is simple to state, but has no simple answer. Caruana and Niculescu-Mizil (2006) carried out an extensive empirical study to compare a number of supervised learning methods based on their performance on empirical datasets. The conclusion from the study was that the performance of a method is essentially dependent on the specific dataset, although the support vector machine (SVM) and random forest (RF) are the top performers for a number of classification tasks. In recent years, Gradient Boosting Machine (GBM), especially one of its variants known as XGBoost (Chen & Guestrin, 2016), and Deep Neural Networks (LeCun, Bengio, & Hinton, 2015; See Chap. 11 for more details) became very popular in the Kaggle community for machine learning competitions (https://www.kaggle.com/competitions) and frequently show up as leading performers. A sensible recommendation for data analysis practitioners may be that they should first try out these methods if they do not already know which one to use.

In addition, given there are many supervised learning methods, it is always tantalizing to think whether some sort of "average" from an ensemble of learners will lead to better predictive performance. Research studies in this direction leads to ensemble learning (Dietterich, 2000). There are two main leading ideas regarding how an ensemble of learners is constructed, one is called bagging (stands for **b**ootstrap **agg**regat**ing**, Breiman, 1996), and the other is called boosting (Friedman, 2001). The bagging primarily aims to reduce the variance of the prediction while the boosting helps to reduce the bias. In each of these approaches, a base learner is identified first, for example, a decision tree. In the bagging approach, multiple training datasets are created through a bootstrapping procedure (Efron & Tibshirani, 1994), and the base learner will be applied to each of the samples. The outputs from the learners from each sample will be combined (average or voting) to improve the predictive performance. In the boosting approach, the ensemble is constructed sequentially, rather than in parallel, as in the bagging approach. A

learner is first trained on the full dataset and subsequent learners are added by fitting the residuals (after applying all preceding learners), by which new learners will assign more weights to the poorly predicted observations by the preceding learners. Random forest and gradient boosting machine (next subsection) are two well-known examples of the bagging and boosting approaches, respectively, both of which use decision trees as learners (e.g., homogeneous learners). In addition to bagging and boosting, there is another ensemble learning approach, known as stacking, which uses the outputs/predictions from several heterogeneous learners (e.g., different machine learning algorithms that do not fall into the same family) as new features and then build a mapping between these features and the labels via another machine learning method. Compared with the bagging and boosting, stacking could reduce both variance and bias. In practice, there are numerous ways to implement stacking, so we (as well as many textbooks on machine learning) primarily focus on bagging and boosting in this chapter.

As the details of each supervised learning algorithms could easily be a Chapter or even a book itself, in the following subsections, we limit ourselves to the basic ideas of some popular methods and refer readers to the textbooks introduced earlier for more rigorous mathematical formulations. We hope that knowing the basic mechanisms behind the methods will help guide practitioners to sensibly choose suitable methods and adjust the corresponding hyperparameters in their applications.

9.2.1 Support Vector Machine

The Support Vector Machine (SVM) was initially introduced in the context of pattern recognition (Vapnik, 1963). It aims at finding a decision hyperplane in the feature space so that data points can be separated into different categories with a maximum margin that is defined as the distance of the closest points (support vectors) from each category to the decision hyperplane. Fig. 9.1 shows a simple case, a two-dimensional feature space, and a binary label, as indicated by the types of the dots, e.g., empty or filled. The left panel shows that there could be multiple ways to draw a straight line (one-dimensional hyperplane) to separate the dots. The right panel shows the straight line that has the maximum margins, which is what SVM is seeking for.

The case shown in Fig. 9.1 is very simple, not only because of the two-dimensional feature space and binary labels but also because the dots are linearly separable (also known as a hard margin). In reality, there could be cases where data of different labels cannot be linearly separated. We show some examples in the left panels of Figs. 9.2 and 9.3. To address these situations, two additional adjustments to SVM have been introduced; one is called regularization and the other is called the kernel trick.

In the case when there is not a linear separation for the dots, as shown in Fig. 9.2, one needs to choose between a decision hyperplane that has a larger margin but more

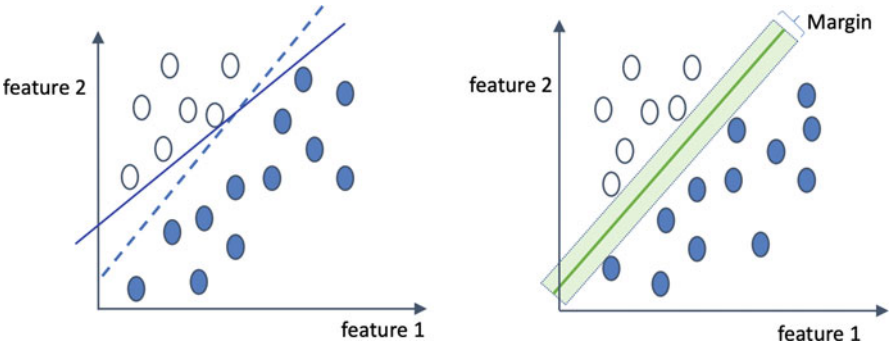


Fig. 9.1 Linearly separable case in a two-dimensional feature space and binary labels. Left: possible hyperplanes. Right: the hyperplane that has a maximum margin. The dots with a green outline are called support vectors

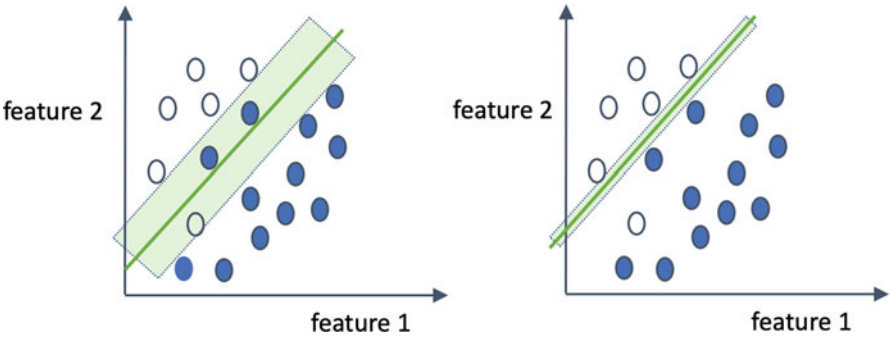


Fig. 9.2 Non-linearly separable cases. Left: larger margin with more misclassified data points. Right: narrower margin with less misclassified data points

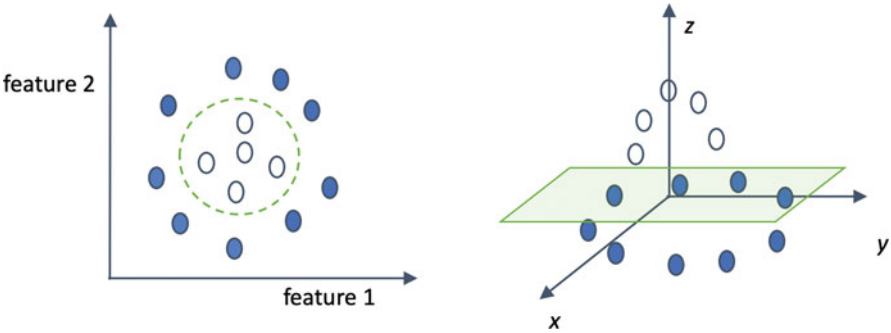


Fig. 9.3 Projection of a 2D feature space to a 3D space to make the data linearly separable

misclassified data points, and the one with a narrower margin but fewer misclassified data points, as illustrated in the left and right panels in Fig. 9.2 respectively. Qualitatively, a hyperplane with a narrower margin may lead to better classification accuracy based on the training data, but is more susceptible to overfitting, while a hyperplane with a larger margin has lower classification accuracy with respect to the training data, but it is more robust against overfitting. In the algorithmic implementation of SVM, a regularization term (equivalent to a soft margin or hinge loss function) is introduced to the loss function, and a hyperparameter (usually denoted as C) is used to adjust between these two preferences quantitatively. Choosing a larger C indicates that one favors a narrower margin with a lower tolerance of misclassified data points. In practice, one needs to decide on the most appropriate C based on the particular nature of the dataset, usually by checking the predictive performance of a range of C based on a certain cross-validation scheme.

On the other hand, as shown in the left panel of Fig. 9.3, though one can apply the above regularization scheme, there could be a better way. For example, one can introduce a set of new variables such that $x = \textit{feature}\ 1$, $y = \textit{feature}\ 2$, $z = (\textit{feature}\ 1)^2 + (\textit{feature}\ 2)^2$. The space spanned by the new variables is now three dimensional, and a 2D plane can be found to separate the empty and filled dots, as shown in the right panel of Fig. 9.3. This transformation of the original feature space into a higher dimensional space to make the data linearly separable is done through a procedure called the kernel trick. The characteristic of data being more likely to be linearly separable when being projected into a higher dimensional space via some non-linear transformation is known as Cover's theorem (Cover, 1965). There are many well-studied kernel functions in SVM, such as the linear kernel and radial basis function (RBF) kernel, and the type, as well as parameters of the kernels become additional hyperparameters of SVM. We will skip the detailed mathematical formulations of the kernels and the algorithms for searching the hyperplanes, and recommend interested readers to the texts on machine learning introduced earlier for more details.

9.2.2 Random Forest

Random forest (Breiman, 2001; Ho, 1995) was introduced based on tree-structured learners, such as decision trees. A decision tree works by forming decision rules on the features recursively to minimize some loss function of the classification. A schematic of the classification process of a binary decision tree classifier is shown in Fig. 9.4. In a decision tree, the root node represents the entire data under consideration. The decision node is where decisions are made to further split the node into sub-nodes. The terminal node, or leaf, represents a node that cannot be split further. To avoid overfitting, a procedure to remove the sub-nodes from decision nodes is often performed after the training, which is called pruning. When implementing a decision tree, an important question that needs to be addressed is which loss function one wants to minimize when creating the decision rules.

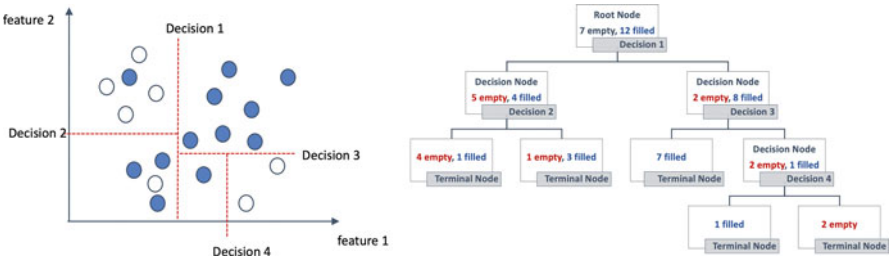


Fig. 9.4 A schematic of decision rules for a binary decision tree classifier. Left is actual decision rules in the 2D feature space. Right is a tree-structured representation

The choice of different loss functions leads to different algorithms for a decision tree. Popular loss functions include the Gini index as used in the CART algorithm (Breiman et al., 1984) and Information Gain as used in the Iterative Dichotomiser 3 algorithm (Quinlan, 1986). After deciding on the loss function, a tree algorithm must also specify a stopping rule, or the condition under which no further splitting will happen. For example, a stopping condition can be set as the minimum leaf size, meaning that if all the nodes have less than a specified threshold, the node splitting process will stop. The stopping rules are usually specified through the hyperparameters in most decision tree algorithm implementations.

One of the main drawbacks of the decision tree is the instability of the results, as a small change in a dataset may lead to quite different decision rules. Therefore, the random forest was proposed to use the "average/voting" of multiple decision trees instead of a single one. Suppose we have N observations and M features in a dataset, then a typical random forest classifier goes as follows. First, create K samples of the dataset using the bootstrapping technique, and each sample has N observations. For each of the K samples, a decision tree is applied. In each of the decision nodes of each of the trees, a decision is made based on the most discriminative one of a randomly chosen F features from the total M features. The number F is usually fixed for all trees in most of the random forest algorithms. The selection of a random subset of features is known as the random subspace method or feature bagging (Ho, 1998), which was proposed to reduce the correlation among the estimators in the ensemble. Overall, the random forest method is much more stable and shows better predictive performance compared to a single decision tree method, though the cost paid for this is the reduced interpretability and increased computation.

9.2.3 Gradient Boosting Machine

When we introduced the SVM and RF, we focused on the classification case because it is easier to convey the basic ideas of the two methods by using as few equations as possible. For the Gradient Boosting Machine (GBM), the regression

case turns out to be easier to convey the key ideas of the method, and some equations are necessary to make the description clear. The basic idea behind GBM is the assumption that the ideal mapping between the dependent and independent variables could be approximated by the sum of functions from the same parametric family. The parameters of these functions will be determined iteratively. First, one fits the first function to the data and determines the corresponding parameters. Then, one determines the second function's parameters by fitting it to the residuals after subtracting the fitting of the first function from the data. Continue this process to obtain the parameters of the other functions until a stopping criterion is met. In the following, we introduce how GBM works in details, largely by following the description in Li (2016) with some notation adjustments.

Suppose we have a dataset that consists of N observations. Each observation is represented by a feature vector $\boldsymbol{X}_i$ and a target value y_i, and we use $(\boldsymbol{X}, y)$ to denote the data collectively. Our goal is to find a mapping function $F(\boldsymbol{X})$ that can minimize the expected value of a loss function, e.g., $E_{X,\,y}L(y, F(\boldsymbol{X}))$. Given that the functional form of $F(\boldsymbol{X})$ can be anything, a practical strategy is usually to restrict $F(\boldsymbol{X})$ as a weighted sum of a parameterized class of functions, such as regression trees. [2] This way, we can write $F\left(\boldsymbol{X}\right) = \sum_{m=0}^{M} h_m\left(\boldsymbol{X}; \boldsymbol{\alpha}_m\right)$, where $h_m(\boldsymbol{X}; \boldsymbol{\alpha}_m)$ is a regression tree, and $\boldsymbol{\alpha}_m$ are the parameters of the regression tree. A procedure to iteratively generate $h_m(\boldsymbol{X}; \boldsymbol{\alpha}_m)$ can go as follows. First, we fit the data $(\boldsymbol{X}, y)$ with a base regression tree (or base learner) $h_0(\boldsymbol{X}; \boldsymbol{\alpha}_0)$. Then, to find a new $h_1(\boldsymbol{X}; \boldsymbol{\alpha}_1)$ that can improve the fitting, we simply fit it to the residuals after substracting the fitted value of $h_0(\boldsymbol{X}; \boldsymbol{\alpha}_0)$ from the target values, e.g., $(\boldsymbol{X}, y - h_0(\boldsymbol{X}; \boldsymbol{\alpha}_0))$. Obviously, $h_0(\boldsymbol{X}; \boldsymbol{\alpha}_0) + h_1(\boldsymbol{X}; \boldsymbol{\alpha}_1)$ will be a better fit to the data than $h_0(\boldsymbol{X}; \boldsymbol{\alpha}_0)$. We can iterate this process to get $h_m(\boldsymbol{X}; \boldsymbol{\alpha}_m)$ by fitting the residual data $(\boldsymbol{X}, y - F_{m-1}(\boldsymbol{X}))$ with $F_{m-1}\left(\boldsymbol{X}\right) \equiv \sum_{k=0}^{m-1} h_k\left(\boldsymbol{X}; \boldsymbol{\alpha}_k\right)$.

In the case of L2 loss function, e.g., $L(y, F(\boldsymbol{X}))\sim \| y - F(\boldsymbol{X})\|^2$, the residuals relate to the negative gradient of the loss function as

$$y - F_m\left(\boldsymbol{X}\right) \propto -\left.\frac{\partial L\left(y, F\left(\boldsymbol{X}\right)\right)}{\partial F\left(\boldsymbol{X}\right)}\right|_{F(\boldsymbol{X})=F_{m-1}(\boldsymbol{X})} \equiv -g_m\left(\boldsymbol{X}\right),$$

Where $g_m(\boldsymbol{X})$ denotes the gradient. Now, our previous procedure for obtaining new $h_m(\boldsymbol{X}; \boldsymbol{\alpha}_m)$ by fitting the residual data $(\boldsymbol{X}, y - F_{m-1}(\boldsymbol{X}))$ can be recast as by fitting $(\boldsymbol{X}, -g_{m-1}(\boldsymbol{X}))$, which is why this method is called gradient boosting. This line of reasoning can be generalized to other forms of loss functions that may obscure the initial residual interpretation but is still effective. Note that the above introduction is not a strict mathematical formulation but rather an intuitive way to show the basic ideas of the method. We refer readers to Friedman (2001) for a more rigorous mathematical treatment.

[2] Regression tree is a generalization of the binary decision tree to the case where the target variable is real-valued instead of binary.

9.3 Hyperparameters

Hyperparameters are those parameters that cannot be estimated through data, such as the regularization and kernel selection in SVM; the number of trees and the size of the feature subspace in RF; and the choice of loss function and the number of boosting stages in GBM. Despite that the choice of hyperparameters often significantly affects the performance of an algorithm, there are actually no fixed rules for choosing the best hyperparameters, though certain default values are assigned to the hyperparameters in most software implementations. Readers need to be aware that these default values are by no means the optimal ones for a particular dataset, and some tuning of the hyperparameters is needed to achieve optimal performance. A brute-force grid search in the hyperparameter space is the most typical approach for deciding the best hyperparameter sets for a given dataset and classification problem. There are many tools or packages for facilitating the hyperparameter search. For example, the Scikit-learn package (Pedregosa et al., 2011) in Python provides functions to run grid search of hyperparameters. Interested readers can find more details from the Scikit-learn documentation website, or from Hao and Ho (2019) for a quick start.

9.4 Examples with Python

In this section, we show some code examples in Python to illustrate how to apply the above machine learning methods in practice. Scikit-learn is an open-source library for machine learning in Python. It includes most of the popular supervised learning algorithms. The general workflow for using a classifier in *Scikit-learn* includes three steps. First, create the model by specifying the hyperparameters of the model. Second, fit the training data to the model and learn the parameters. Finally, apply the fitted model to the test data to get predicted labels. The pseudo-code for these steps is as follows, where `classifier()` is an instance of one of the supervised learning methods. X is the feature matrix whose rows index observations and columns index different feature variables. y is the array of labels with each element as the corresponding label value.

```
model = classifier(hyperparameters = something)
model.fit(X_train, y_train)
y_test = model.predict(X_test)
```

The data used in this example are the labeled chats from an online collaborative problem-solving (CPS) study (Hao et al., 2015). Each turn of the chats has been coded into one of four categories of CPS skills based on a coding rubric (Liu et al., 2015). The goal is to build an automated classifier using machine learning methods to label the chats automatically. Natural language processing (NLP) techniques (see Chap. 14) have been applied to convert each turn of the chats into a numerical vector (for simplicity, we consider only the uni-gram and bigram). In the following Fig. 9.5,

we show the Python code for building an automated classifier based on the above dataset by using SVM, RF, and GBM. The detailed meaning of the codes can be found in the corresponding comments of each cell. It is worth noting that only the default values of the hyperparameters in each method have been used for illustration purposes. Even without any further tuning, the three supervised learning methods still generate reasonably good classification accuracy (defined as the total number of agreements divided by the total number of observations) of about 0.7, which is far better than the baseline (e.g., random assignment of the most frequent category) of about 0.3.

1. Load the needed libraries

```
In [1]: from sklearn.svm import LinearSVC # for Support Vector Machine
        from sklearn.ensemble import RandomForestClassifier # for Random Forest
        from sklearn.ensemble import GradientBoostingClassifier # for Gradient Boosting
        Machine

        from sklearn.model_selection import train_test_split
        from sklearn.metrics import accuracy_score
        import pandas as pd
        import warnings
        warnings.simplefilter('ignore')
```

2. Instantiate the models and check the details

```
In [2]: model_SVM = LinearSVC()
        model_RF = RandomForestClassifier()
        model_GBM = GradientBoostingClassifier()

In [3]: model_SVM

Out[3]: LinearSVC(C=1.0, class_weight=None, dual=True, fit_intercept=True,
                  intercept_scaling=1, loss='squared_hinge', max_iter=1000,
                  multi_class='ovr', penalty='l2', random_state=None, tol=0.0001,
                  verbose=0)

In [4]: model_RF

Out[4]: RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                               max_depth=None, max_features='auto', max_leaf_nodes=None,
                               min_impurity_decrease=0.0, min_impurity_split=None,
                               min_samples_leaf=1, min_samples_split=2,
                               min_weight_fraction_leaf=0.0, n_estimators='warn',
                               n_jobs=None, oob_score=False, random_state=None,
                               verbose=0, warm_start=False)

In [5]: model_GBM

Out[5]: GradientBoostingClassifier(criterion='friedman_mse', init=None,
                                   learning_rate=0.1, loss='deviance', max_depth=3,
                                   max_features=None, max_leaf_nodes=None,
                                   min_impurity_decrease=0.0, min_impurity_split=None,
                                   min_samples_leaf=1, min_samples_split=2,
                                   min_weight_fraction_leaf=0.0, n_estimators=100,
                                   n_iter_no_change=None, presort='auto',
                                   random_state=None, subsample=1.0, tol=0.0001,
                                   validation_fraction=0.1, verbose=0,
                                   warm_start=False)
```

Fig. 9.5 Implementation of the SVM, RF, and GBM in the Scikit-learn package in Python programming language

3. Read in the data file

```
In [6]: X=pd.read_csv('chat_bigram_feature.csv').values
        y=pd.read_csv('chat_label.csv').values.ravel()
```

4. Create the training and validation sets

```
In [7]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5, random_state
        =42)
```

5. Train the models and generate predicted labels

```
In [8]: # train the models with training set
        model_SVM.fit(X_train,y_train)
        model_RF.fit(X_train,y_train)
        model_GBM.fit(X_train,y_train)

        # -- get the predicted labels on the test dataset.
        y_pred_SVM = model_SVM.predict(X_test)
        y_pred_RF = model_RF.predict(X_test)
        y_pred_GBM = model_GBM.predict(X_test)
```

6. Check the predictive performance

```
In [9]: # calculate the accuracy of the predicted labels from SVM
        accuracy_score(y_pred_SVM, y_test).round(3)
Out[9]: 0.692

In [10]: # calculate the accuracy of the predicted labels from Random Forest
         accuracy_score(y_pred_RF, y_test).round(3)
Out[10]: 0.644

In [11]: # calculate the accuracy of the predicted labels from Gradient Boosting Machine
         accuracy_score(y_pred_GBM, y_test).round(3)
Out[11]: 0.642
```

Fig. 9.5 (continued)

The example code snippet shows the general idea of building an automated text classifier using machine learning methods. Readers should be aware that a real automated annotation engine requires much more effort, such as NLP feature engineering, machine learning model selection, hyperparameter tuning, and cross-validation. Interested readers should consult relevant texts and references. For example, a good starting point is Hao et al. (2017) and references therein.

9.5 Summary

In this Chapter, we introduced supervised machine learning and its different emphasis as compared to the statistical inference that is more familiar to researchers with statistic and psychometric backgrounds. By this, we hope readers can make sensible decisions regarding which approaches are best suited for their application. We introduced the basic ideas behind three popular supervised learning methods, the SVM, RF, and GBM. We further showed how to apply these methods to a

real-world problem using coding examples written in Python. As stated earlier, the goal of this Chapter is not to provide comprehensive coverage of supervised learning but more a gentle introduction to provide researchers with a general sense of supervised learning and how to use it in practice. We encourage interested readers to check out more details about supervised learning from those dedicated volumes as suggested in the introduction. In the next chapter (Chap. 10), a similar introduction to unsupervised machine learning will be provided.

Acknowledgement The author thanks Mr. Andrew Cantine and Robert Mislevy for helpful suggestions and copyediting.

References

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). *Classification and regression trees*. Wadsworth.

Breiman, L. (1996). Bagging predictors. *Machine Learning, 24*(2), 123–114.

Breiman, L. (2001). Random forests. *Machine Learning, 45*(1), 5–32.

Caruana, R., & Niculescu-Mizil, A. (2006). An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning* (ICML '06; pp. 161–168). New York, NY: Association for Computing Machinery.

Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 785–794). ACM.

Cover, T. M. (1965). Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Transactions on Electronic Computers, 3*, 326–334.

Dietterich, T. G. (2000, June). Ensemble methods in machine learning. In *International workshop on multiple classifier systems* (pp. 1–15). Springer.

Efron, B., & Tibshirani, R. J. (1994). *An introduction to the bootstrap*. CRC press.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 1189–1232.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction*. Springer.

Hao, J., & Ho, T. K. (2019). Machine learning made easy: A review of Scikit-learn package in Python programming language. *Journal of Educational and Behavioral Statistics, 44*(3), 348–361.

Hao, J., Liu, L., von Davier, A., & Kyllonen, P. (2015). Assessing collaborative problem solving with simulation based tasks, proceeding of 11th international conference on computer supported collaborative learning. Gothenburg, Sweden.

Hao, J., Chen, L., Flor, M., Liu, L., & von Davier, A. A. (2017). CPS-rater: Automated sequential annotation for conversations in collaborative problem-solving activities. ETS Research Report No. RR-17-58. Princeton, NJ: Educational Testing Service.

Ho, T. K. (1995, August). Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition* (Vol. 1, pp. 278–282). IEEE.

Ho, T. K. (1998). The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 20*(8), 1–22.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436.

Li, C. (2016). A gentle introduction to gradient boosting. URL: http://www.ccs.neu.edu/home/vip/teach/MLcourse/4_boosting/slides/gradient_boosting.pdf

Liu, L., Hao, J., von Davier, A., Kyllonen, P., & Zapata-Rivera, D. (2015). A tough nut to crack: Measuring collaborative problem solving. In Y. Rosen, S. Ferrara, & M. Mosharraf (Eds.), *Handbook of research on computational tools for real-world skill development*. Hershey, PA: IGI-Global.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research, 12*, 2825–2830.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning, 1*, 81–106.

Samuel, A. L. (1959). Some studies on machine learning using the game of checkers. *IBM Journal of Research and Development, 3*, 211–229.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction*. MIT Press.

Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2016). *Data mining: Practical machine learning tools and techniques*. Morgan Kaufmann.

Vapnik, V. (1963). Pattern recognition using generalized portrait method. *Automation and Remote Control, 24*, 774–780.

Chapter 10
Unsupervised Machine Learning

Pak Chung Wong

Abstract The chapter introduces the concept of machine learning with an emphasis on unsupervised learning algorithms and applications. The discussion starts with a brief background on machine learning and then a high-level discussion on the differences between supervised and unsupervised learning algorithms. We present three categories of unsupervised machine learning techniques that include clustering, outlier detection, and dimension reduction; five prevailing unsupervised learning algorithms that include K-means, agglomerative clustering, DBSCAN, principal component analysis, and multidimensional scaling; and five Python programming examples that demonstrate the learning concepts and results using psychometric assessment data collected from an online collaborative problem-solving environment. This chapter demonstrates the potential of machine learning and highlights the opportunities it presents in psychometric research and development.

10.1 Introduction

Arthur Lee Samuel (Samuel, 1959), who coined the term "machine learning," defined the then new discipline as a sub-field of Computer Science that gives "computers the ability to learn without being explicitly programmed." Machine learning (Machine Learning, 2017) has since found its way into different data science communities, which include, among others, exploratory data analysis (Tukey, 1997), data mining (Fayyade t al., 1996), and visual analytics (Wong & Thomas, 2004). More recently, (Pedro Domingos, 2015) broke down the five tribes (or paradigms) of machine learning as symbolists, connectionists, evolutionaries,

The R or Python codes can be found at the GitHub repository of this book: https://github.com/jgbrainstorm/computational_psychometrics

P. C. Wong (✉)
University of Iowa, Iowa City, IA, USA

© Springer Nature Switzerland AG 2021

A. A. von Davier et al. (eds.), *Computational Psychometrics: New Methodologies for a New Generation of Digital Learning and Assessment*, Methodology of Educational Measurement and Assessment, https://doi.org/10.1007/978-3-030-74394-9_10

Bayesians, and analogizers. Signature technologies from each of these tribes are expert systems, deep learning, genetic algorithms, Bayesian hierarchical modeling, and cluster analysis respectively.

We look into machine learning from a data science perspective of whether the underlying algorithms are supervised or unsupervised methodologies. In particular, this chapter focuses on the unsupervised learning techniques as applied to computational psychometrics studies. For example, we use a clustering algorithm to better understand the common behavior shared by a group of learners in a computational psychometrics assessment study. Our discussion uses layman's terms to explain the design approaches of individual learning approaches. Graphical illustrations are used extensively to supplement the discussion of the algorithmic concept instead of relying solely on the use of notation to explain the theory behind. The chapter serves as an introduction to machine learning in computational psychometrics with programming examples.

10.2 Supervised Versus Unsupervised Learning

Supervised learning (Supervised Learning, 2017) is about predicting a specific quantity, such as whether an incoming email is spam or ham. The supervised learning process always requires training examples with labels, and enough examples to make the learning algorithm statistically reliable. A label in this context could mean an informative tag or class, such as a car or a building, which is meaningful to know in the learning process. The accuracy of the learning result from a supervised learning process can be measured. For example, in the above spam or ham example, a human can always inspect the email under study and verify the accuracy of the prediction.

Unsupervised learning (Unsupervised Machine Learning, 2017), on the other hand, is about understanding the data, such as looking for unusual structures like outliers or clusters. It is never about looking for something specific, like the above email example in supervised learning. Unsupervised learning does not require training data, or data with labels in the learning process. Results of unsupervised learning are evaluated either qualitatively (e.g., does it enhance our understanding of the data?) or indirectly (e.g., does it improve the performance results of the other algorithms applied to the learning results?).

This chapter focuses on three main classes of unsupervised learning algorithms: (1) clustering, (2) outlier detection, and (3) dimension reduction. Readers are directed to Chap. 9 of this book for another discussion on supervised machine learning.

10.2.1 Clustering

Clustering (Xu & Wunsch, 2008) is mostly an unsupervised learning task that discovers interesting population-related structures in the underlying data. We use clustering to mine information such as: (1) number of clusters, (2) cluster sizes, (3) common properties among clusters, (4) further splitting up into sub-sub-populations and beyond, and, sometimes, (5) outliers. Our discussion focuses on three clustering techniques, which include K-means, hierarchical clustering, and DBSCAN (Ester et al., 1996).

Today's clustering techniques are often categorized in three ways: (1) monothetic vs. polythetic, (2) divisive (or partition) vs. agglomerative, or (3) hard vs. soft clustering.

1. Monothetic techniques identify clusters based on one dimension of a multidimensional dataset, whereas polythetic techniques are based on all dimensions. An example of a monothetic clustering algorithm is k-d Tree (Bentley, 1975). The vast majority of unsupervised clustering applications found in the literature today as well as the examples in this chapter are polythetic.
2. Divisive techniques split instances of a dataset from one large group into smaller groups. Agglomerative techniques start with an individual instance in its own cluster and combine multiple clusters into larger and larger groups. The top-down-based K-means is a divisive technique. The bottom-up-based hierarchical clustering is an agglomerative technique. The density-based DBSCAN is more related to agglomerative clustering in practice.
3. In hard clustering, a data instance can be a member of one and only one cluster. In soft clustering, a data instance can be a member of multiple clusters with different degrees (or strengths) of association. Techniques described in this chapter belong to the hard clustering category. An example of a soft clustering technique is Expectation-Maximization (EM) (Dempster et al., 1977), which uses a probabilistic model for representing the presence of subpopulations within an overall population (such as a Gaussian mixture model) to cluster the data.

The definition of a cluster is more abstract in nature than many other metrics in life. For example, one can see either two or three clusters in Fig. 10.1, depending on (1) the definition of a cluster, or (2) the number of clusters that you want to see. The same holds true in most of the prevailing machine learning algorithms in the literature. For example, among the three clustering algorithms described later in this chapter, both K-means and hierarchical clustering require the number of clusters as part of the learning input, whereas DBSCAN needs a precise definition of a cluster, namely, its density (or the number of instances/data points within a predefined space).

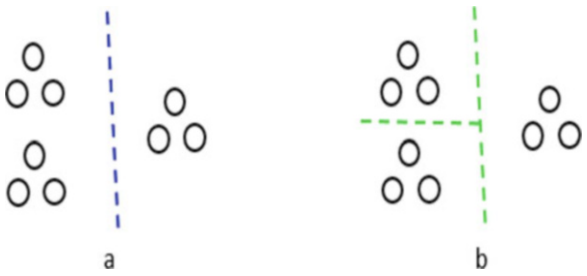


Fig. 10.1 An illustration of clustering showing either two (**a**) or three (**b**) clusters

10.2.2 Outlier Detection

The concepts behind outliers and anomalies are very similar in terms of the statistics behind them. In machine learning, an outlier in a dataset is an instance that is not like the others, whereas an anomaly is an instance that is not acting as it used to. Thus, they need different learning techniques to detect their presence.

For outlier detection, we compare data that should be behaving similar to one another. For anomaly detection, we use the data's past history to see whether it is deviating from where we think it should be. Although the nomenclature of these data types is not standardized in the industry, our outlier detection discussion here focuses on the detection of one of those things that is not like the others within a dataset. The unsupervised learning technique that we described to detect outliers is DBSCAN, which was first introduced in our discussion as an unsupervised learning technique for clustering.

10.2.3 Dimension Reduction

In machine learning and data mining, the goal of dimension reduction is to represent the same instances of a multidimensional dataset using fewer data dimensions (or variables), often through the process of either feature selection (Feature Selection, 2017) or feature extraction (Feature Extraction, 2017).

In feature selection, we decide which variables are important and select them for the analysis. In Fig. 10.2a, we select only three dimensions (i.e., X_1, X_3, X_N) from an N-dimensional (ND) dataset (i.e., $X_1, X_2 \ \ldots \ X_N$) based on our knowledge of the data.

In feature extraction, we build a new set of dimensions $Y_1, Y_2 \ \ldots \ Y_n$ to represent the same instances of an ND dataset with dimensions $X_1, X_2 \ \ldots \ X_N$, where $N \gg n$. As illustrated in Fig. 10.2b, each of the $Y_{j,}$ dimensions represent a combination of portions of all the original dimensions $X_1, X_2 \ \ldots \ X_N$. The extraction process attempts to preserve as much discriminative structure of the original dimensions $X_1, X_2 \ \ldots \ X_N$ as possible. For example, in using Principal Component Analysis (PCA) that will be discussed later in this chapter, we want to preserve most, but not

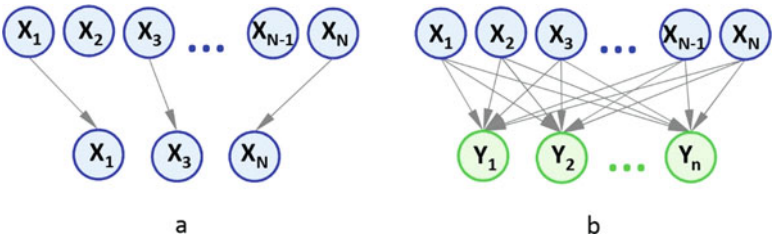


Fig. 10.2 (**a**) Feature selection and (**b**) feature extraction in dimension reduction

all of the "variance" (i.e., structure) within the data, thereby reducing the amount of information (data variables and thus data size) but retaining the relevant information necessary to represent the original data.

10.2.4 Python Programming Examples

Computer programming examples presented in this chapter are coded in an Anaconda (Anaconda, 2017) version of Python 3.6 (Python, 2017) using the Jupyter Notebook App (Jupyter, 2017) and Python modules that include Counter, DBSCAN, Agglomerative Clustering, Decomposition, dendrogram, linkage, matplotlib, numpy, pandas, PCA, scipy, and sklearn. R (R, 2017) and Ruby (Ruby, 2017) are two other popular programming language options for developing similar machine learning algorithms and/or applications.

For those who are interested in practicing machine learning but do not want to write their own code, WEKA (Weka, 2017) provides a friendly workbench that incorporates many standard learning techniques ready for everyday machine learning challenges. WEKA is licensed under the GNU General Public license and thus is free to download and use.

10.2.5 Psychometric Assessment Dataset for Demonstration

We use a psychometric assessment dataset (Polyak et al., 2017) to demonstrate the usage of the Python programs presented in this chapter. The data was "collected from a sample of middle school children who interacted with a game-like, online simulation of collaborative problem-solving tasks." The five data variables used in this discussion are Feature Identification (FI), Maintaining a Shared Understanding (MU), Engagement/Interaction (EN), Strategy (S), and Evaluate (EV). There are 159 instances in the demo dataset. The collection and analysis of the above five non-cognitive learning assessment qualities in the demo dataset is a part of an ongoing R&D project known as Collaborative Problem Solving (CPS).

10.3 Unsupervised Machine Learning Techniques

There is an evident technological commingling among today's machine learning algorithms, in which they borrow ideas from each other to address similar computational and/or analytical needs. This section is thus organized based on individual unsupervised learning techniques (instead of technique categories), as some of them address more than one category of learning algorithms described earlier.

10.3.1 K-Means

K-means is a polythetic, divisive clustering technique with hard cluster boundaries. Given a multidimensional dataset, we ask K-means to find K clusters from the dataset and K-means will return the exact number of clusters. In other words, the K-Means algorithm does not suggest an optimal K-value.

10.3.1.1 Primary Concept of K-Means

Given a constant k and a set of data points in an n-D space R^n, K-means will:

1. Randomly assign k locations in R^n as initial cluster centroids $c_1 \ldots c_k$. A centroid c of a finite set of m points $x_1, x_2 \ldots x_m$ is $c = \frac{(x_1+x_2+\cdots+x_m)}{m}$, which minimizes the sum of squared Euclidean distances between itself and each point in the set
2. For each point x_i, find the nearest centroid c_j and then assign x_i to cluster j
3. For each cluster j, calculate a new centroid c_j using the points assigned to cluster j
4. Repeat steps (2) and (3) until convergence (i.e., until none of the cluster assignments change).

Here we use a simple 2D dataset with 14 data points shown as empty circles in Fig. 10.3a to demonstrate the concept of the K-means algorithm. We set $k = 2$ (i.e., dividing the 14 points into two clusters) in this example. The two randomly assigned centroids are shown as blue and green crosses in Fig. 10.3a.

The next step is to calculate the minimum distances between all the points and the two centroids. Because we use Euclidean distance as a distance metric, there exists a bisecting line (shown as the red dash line in Fig. 10.3b) such that every point on the left of the dash line goes to the blue cluster and the rest of the points to the green cluster.

After the blue and green clusters are identified, K-means updates the cluster centroids by recalculating two new centroids for each cluster using the points assigned to them, as shown in Fig. 10.3c.

The uncovering of new centroids repeats (from Fig. 10.3d to Fig. 10.3f) until convergence—meaning either the Euclidean distance between two consecutive centroids is less than a pre-defined value or the cluster members do not change.

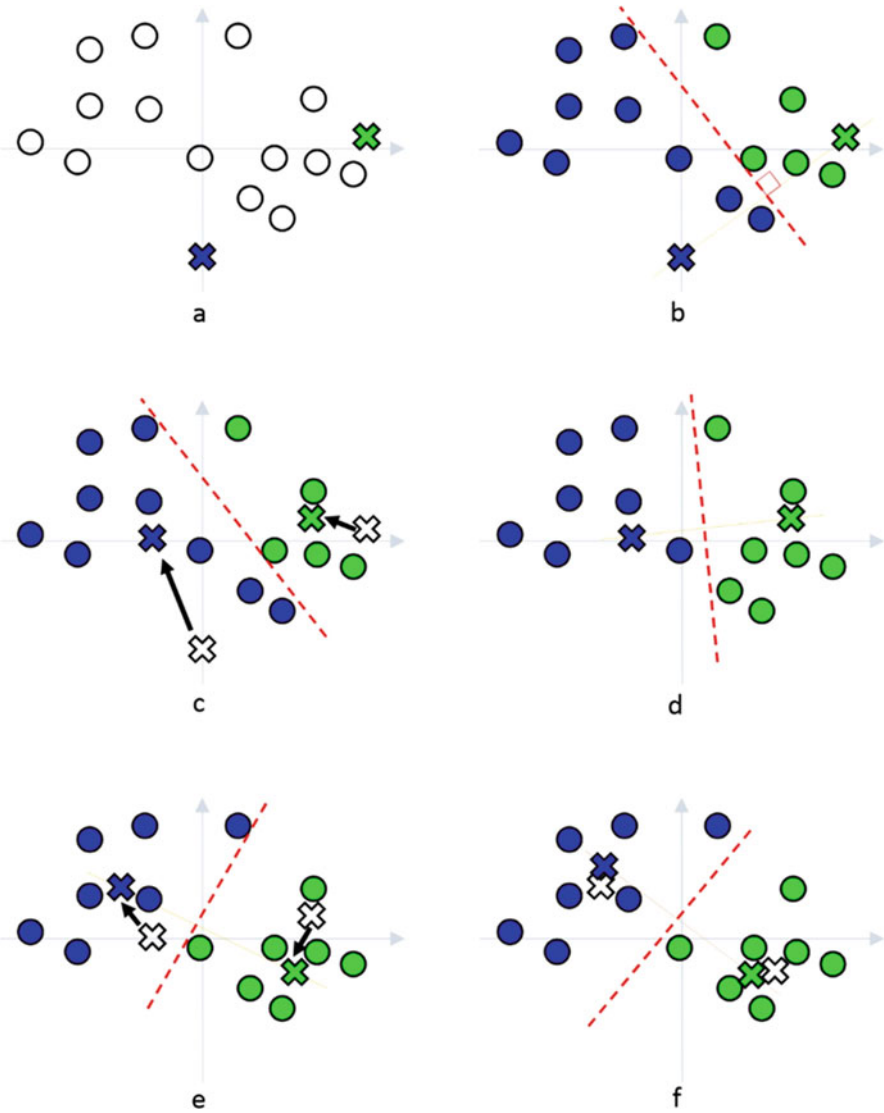


Fig. 10.3 (**a**) Two randomly assigned centroids are shown as the blue and green crosses. (**b**) The data points on the left of the dash lines go to the blue cluster and the rest go to the green cluster. (**c**) New centroids for the blue and green clusters are re-calculated based on the data points in the corresponding clusters. (**d**) Another bisecting line is drawn and new clusters are assigned. (**e**) Another iteration of re-calculating centroids and re-clustering. (**f**) Members of the blue and green clusters do not change and the iteration stops

The results are shown in Fig. 10.3f where the two clusters are depicted in blue and green respectively.

10.3.1.2 K-Means Demo Program Using Python

Figure 10.4 shows the K-means clustering algorithm as implemented using Python 3. Lines 1–4 set up the Python modules required for this program. Lines 6–7 request inline figure output and define figure sizes. Lines 9–10 input data from 'cps.csv' file and create a pandas dataframe. Line 12 calls KMeans() to create three clusters. Lines 14–21 format and present a K-Means clustering visualization of three clusters as shown in the bottom of Fig. 10.4.

10.3.2 Hierarchical Clustering

Our discussion of hierarchical clustering focuses on the bottom-up agglomerative algorithm, which merges nearby clusters iteratively into bigger clusters. The algorithm's approach is to ensure that nearby points end up in the same cluster. Because we are dealing with clusters of data points instead of the data points themselves in this algorithm, we first define the meaning of the distance between two clusters.

Given cluster $X = \{x_1, x_2 \ \ldots \ x_a\}$, cluster $Y = \{y_1, y_2 \ \ldots \ y_b\}$, and $d(x_i, y_j)$ = distance between $x_i \in X$ and $y_j \in Y$, the distance between clusters X and Y can be defined as one of the following:

1. Single linkage: $d_{XY} = min_{i,j}\, d(x_i, y_j)$, the shortest pairwise distance between $x_i \in X$ and $y_j \in Y$
2. Complete linkage: $d_{XY} = max_{i,j}\, d(x_i, y_j)$, the longest pairwise distance between $x_i \in X$ and $y_j \in Y$
3. Average linkage: $d_{XY} = \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b} d\left(x_i, y_j\right)$, the average of all pairwise distances of members of clusters X and Y
4. Centroids: $d_{XY} = d\left(x, y\right)$, the distance between the centroids of X and Y
5. Ward's method: This method does not involve any distance measures. At each iteration, the algorithm merges the two clusters that result in the smallest increase in the total sum of squares of the distances of each point to its cluster centroid.

10.3.2.1 Primary Concept of Agglomerative Clustering

First, we describe the agglomerative clustering algorithm and then explain it using a simple 2-D example. Given a set of data points $\{x_1, x_2 \ \ldots \ x_m\}$ in R^n, the algorithm will:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans

% matplotlib inline
plt.figure(figsize=(4, 4))

cps = pd.read_csv('cps.csv', header=None)
X = cps.iloc[:,:].values

clustering = KMeans(n_clusters=3, random_state=8).fit(X)

cps_df = pd.DataFrame(cps)
cps_df.columns = ['FI','MU','EM', 'EV', 'S']
color_theme = np.array(['red','green','blue'])
plt.scatter(x=cps_df.FI, y=cps_df.EM,
            c=color_theme[clustering.labels_], s=30)
plt.xlabel(cps_df.columns[0], fontsize=12)
plt.ylabel(cps_df.columns[2], fontsize=12)
plt.show()
```

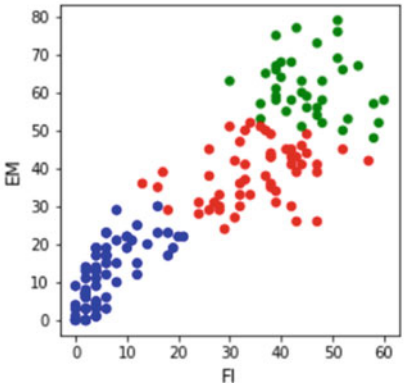


Fig. 10.4 A Python program that uses the K-Means algorithm to cluster the data and creates a scatterplot of three clusters in blue, red, and green

1. Start with a collection of singleton clusters $C = \{c_1, c_2 \ \ldots \ c_m\}$ such that each cluster c_i contains one of the m data points, i.e., $c_i = \{x_i\}$
2. Find the closest cluster pairs c_i and c_j from C using one of the distance measures described earlier
3. Merge the cluster pairs c_i and c_j into c_{i+j}
4. Add c_{i+j} to C, remove c_i and c_j from C
5. Repeat steps (2) to (4) until there is only one cluster left.

Graphically, Fig. 10.5a depicts 13 2D data points, which are initially treated as 13 clusters as described earlier. Using the single linkage distance metric, we identify that the closest cluster pairs are {A} and {B}, and subsequently merge them into {A,

B} in Fig. 10.5b. The blue numbers in Fig. 10.5 denote the order of merging in this example. The process iterates twice in Fig. 10.5c where {J} and {K} followed by {C} and {D} are merged. In Fig. 10.5d, the fourth merge takes place between two previously merged clusters {A, B} and {C, D}, which form a new cluster {A, B, C,

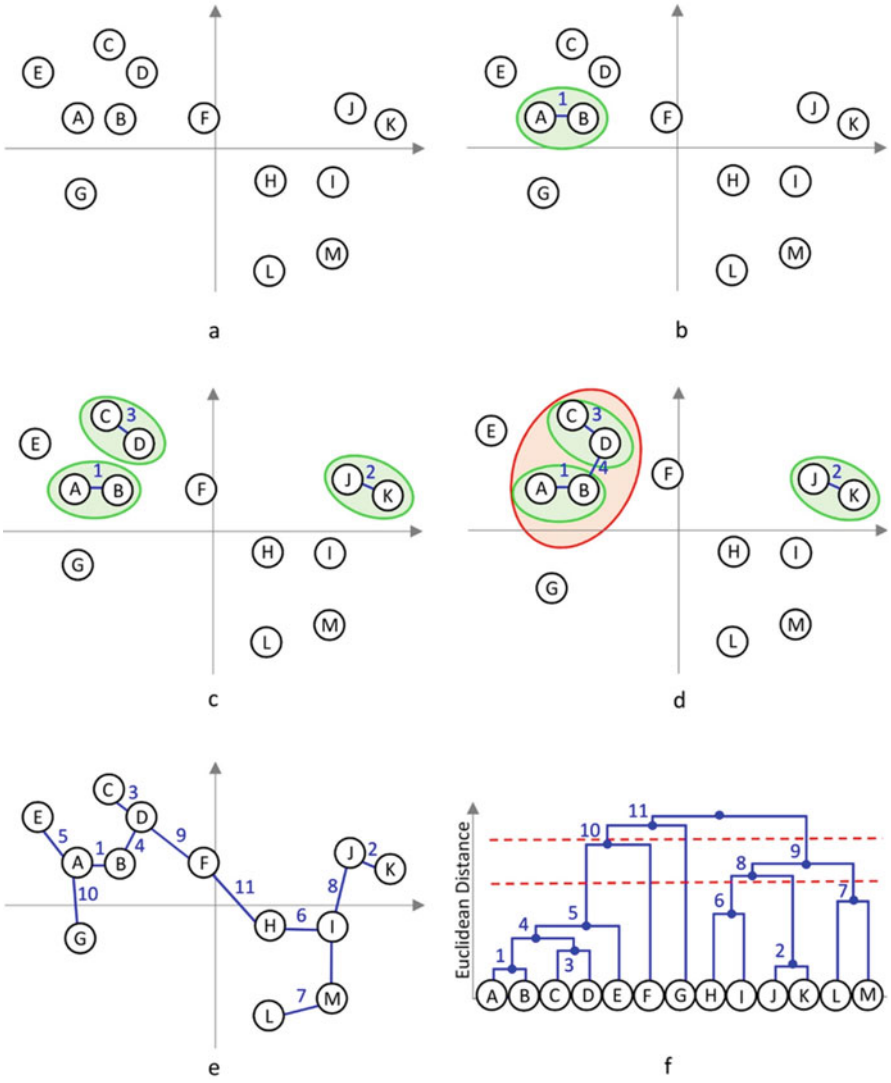


Fig. 10.5 (**a**) A plot of 13 2D data points treated as 13 clusters. (**b**) The closest cluster pairs {A} and {B} are merged. (**c**) {J} and {K} are merged followed by {C} and {D}. (**d**) {A, B} and {C, D} are merged. (**e**) The merges happen in the order of the blue number. (**f**) A dendrogram representation of the agglomerative clustering

D}. The process continues in Fig. 10.5e, where the merges happen in the order of the blue number, until all 13 data points are merged into one cluster.

The clustering results in Fig. 10.5e can be plotted as a dendrogram as depicted in Fig. 10.5f. The y-axis of the dendrogram represents the pairwise Euclidean distance of the initial clusters, which are listed along the x-axis in the same order as the merging order shown in Fig. 10.5e. Moving the red dash lines up and down along the y-axis direction in Fig. 10.5f can reveal the number of clusters and their corresponding members. For example, the upper dash line shows three clusters (i.e., {A, B, C, D, E, F}, {G}, {H, I, J, K, L, M}) whereas the lower one shows six (i.e., {A, B, C, D, E}, {F}, {G}, {H, I}, {J, K}, {L, M}).

10.3.2.2 Agglomerative Clustering Demo Program Using Python

Figure 10.6 shows the agglomerative clustering program in Python 3. Similar to the K-Means program in Fig. 10.4, lines 1–4 set up modules that are required for the agglomerative algorithm to run. Lines 6–8 request inline figure output and define figure style and sizes. Lines 10–11 input data from the 'cps.csv' file and return a list of values. Line 12 asks for the Ward's minimum variance approach (Ward's Method, 2017) to identify the clusters. Line 14 creates the dendrogram, which is printed out by lines 17–21 as shown in Fig. 10.6.

Lines 19 and 20, which draw the two dotted horizontal lines at distances 150 and 250 on top of the dendrogram in Fig. 10.6, create three ({6, 16, 13, 14, 7, 18}, {12, 9, 7}, and {29, 15, 13}) and two ({6, 16, 13, 14, 7, 18}, {12, 9, 7, 29, 29, 15, 13}) clusters respectively.

10.3.3 DBSCAN

DBSCAN (Ester et al., 1996) is a polythetic, density-based clustering technique with hard cluster boundaries. Unlike the K-means algorithm that requires the number of clusters to start the learning process, DBSCAN asks for two different pieces of information: the maximum radius of a neighborhood (shown as Eps in Fig. 10.7a) and the minimal number of points (or MinPts) inside the Eps-bounded neighborhood in the same figure. Assuming Eps = 1 cm and MinPts = 5, the blue point C in Fig. 10.7b is called a *core* point because its Eps-bounded neighborhood has five member points within its boundary. When a point does not meet the MinPts requirement but has at least one core point located inside its Eps-bounded neighborhood, it is called a *border* point. Figure 10.7c depicts a border point B in green, which has a core point C but only three member points within its neighborhood. The points that are neither a core nor a border point are called *Noise* points, such as point N in Fig. 10.7d.

```
import pandas as pd
from scipy.cluster.hierarchy import dendrogram, linkage
import matplotlib.pyplot as plt
from sklearn.cluster import AgglomerativeClustering

%matplotlib inline
plt.style.use('seaborn-whitegrid')
plt.figure(figsize=(8, 4))

cps = pd.read_csv('cps.csv', header=None)
X = cps.iloc[:,:].values
Z = linkage(X, 'ward')

dendrogram(Z, truncate_mode='lastp', p=12, leaf_rotation=-45.,
           leaf_font_size=12., show_contracted=True)

plt.xlabel('Cluster Size', fontsize=12)
plt.ylabel('Distance', fontsize=12)
plt.axhline(y=250, linestyle='dashed')
plt.axhline(y=150, linestyle='dotted')
plt.show()
```

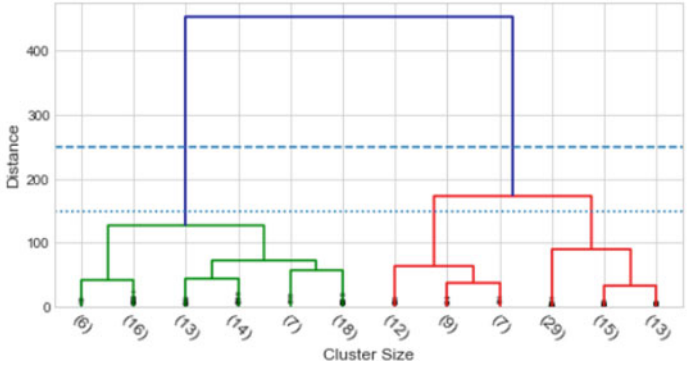


Fig. 10.6 A Python program that uses the agglomerative clustering algorithm to cluster the data and creates a dendrogram

10.3.3.1 Primary Concept of DBSCAN

After all the DBSCAN terminologies, which include Eps, MinPts, Core, Border, and Noise, are properly defined and established, the DBSCAN algorithm is straightforward to implement. The goal is to assign each core and border point to a cluster. Here we use a simple dataset with 12 data points as depicted in Fig. 10.8a to illustrate the concept.

1. Pick an unexplored core point in the dataset such as the blue X point in Fig. 10.8a
2. Perform a breadth-first search from X and establish a neighborhood bounded by the black dash line as shown in Fig. 10.8b
3. Iteratively apply the search on each unexplored core points in the above neighborhood, as shown in Fig. 10.8c and later in Fig. 10.8d.

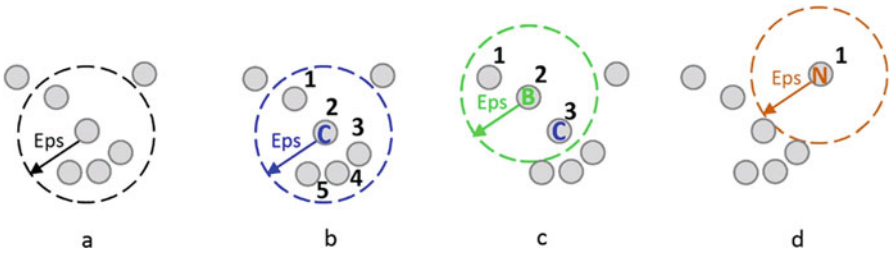


Fig. 10.7 (**a**) A circular area defined by Eps. (**b**) C is a core point because it meets the MinPts = 5 requirement. (**c**) B is a border point because it has a core point in its neighbourhood but does not meet the MinPts requirement. (**d**) N is a noise point as it fails to meet the requirements of both a core point and a border point

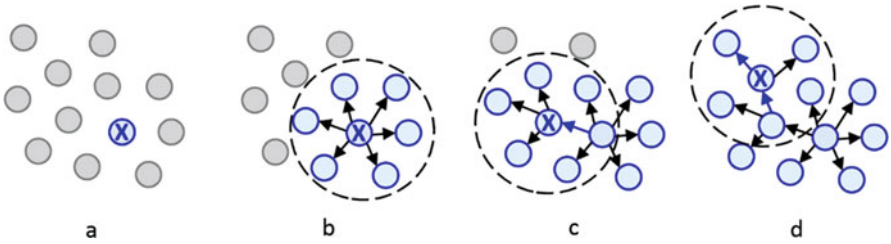


Fig. 10.8 (**a**) The blue highlighted point is an unexplored core point in this example. (**b**) A local search within a circular area as defined by Eps. (**c**) The search is expanded to one of the neighboring core point highlighted in blue. (**d**) The search stops when all the core and boarder points are explored

The resulting tree-structure shown in Fig. 10.8d form a point cluster. The DBSCAN algorithm repeats the search until all core and border points have been assigned to a cluster.

10.3.3.2 DBSCAN Demo Program Demo Using Python

Figure 10.9 shows the DBSCAN program implemented in Python 3. Similar to the previous Python programs, lines 1–6 set up modules that are required for the DBSCAN algorithm to run. Lines 8–9 request inline figure output and define figure sizes. Lines 11–12 input data from the 'cps.csv' file and create a list of data values. Line 14 identifies the clusters using Eps = 8.0 and MinPts (or min_samples in the Python program) = 8. Line 15 outputs the results of the DBSCAN call. Lines 17–23 create the scatterplot as shown in the bottom of Fig. 10.10.

The fit() function call in line 14 of Fig. 10.9 gives cluster labels (in sequential order 0, 1, 2 . . .) for each point in the cps.cvs dataset. Noise data points (or outliers) are given the label −1. Line 15 outputs the contents of the Counter() dictionary, suggesting that there are 67 noise data points, 61 points in cluster 0, and 31 points

```
1  import numpy as np
2  import pandas as pd
3  import matplotlib.pyplot as plt
4  import matplotlib.cm as cm
5  from sklearn.cluster import DBSCAN
6  from collections import Counter
7
8  %matplotlib inline
9  plt.figure(figsize=(4, 4))
10
11 cps = pd.read_csv('cps.csv', header=None)
12 X = cps.iloc[:,:].values
13
14 model = DBSCAN(eps=8.0, min_samples=8).fit(X)
15 print(Counter(model.labels_))
16
17 cps_df = pd.DataFrame(cps)
18 cps_df.columns = ['FI','MU','EM', 'EV', 'S']
19 color_theme = cm.rainbow(np.linspace(0,1,len(Counter(model.labels_))))
20 plt.scatter(x=cps_df.FI,y=cps_df.EM,c=color_theme[model.labels_],s=30)
21 plt.xlabel(cps_df.columns[0], fontsize=12)
22 plt.ylabel(cps_df.columns[2], fontsize=12)
23 plt.show()
```

```
Counter({-1: 67, 0: 61, 1: 31})
```

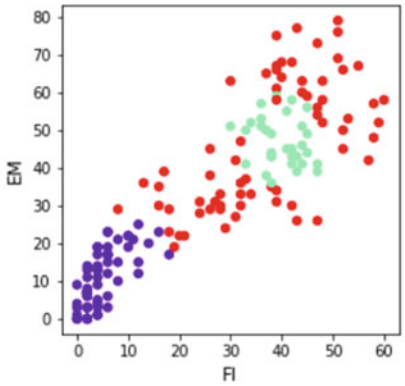


Fig. 10.9 A Python program that uses the DBSCAN algorithm to cluster the data and identify noise data points

in cluster 1 when using eps = 8.0 and min_samples = 8 in the learning process. The K-means program in Fig. 10.4 explicitly asks for three clusters (i.e., k = 3, line 12 of Fig. 10.4) and thus we could pre-defined three different colors (red, green, and blue; line 17 of Fig. 10.4) to show the clustering results in Fig. 10.4. On the other hand, the DBSCAN program in Fig. 10.9 would not know the exact number of clusters until after the algorithm is executed (line 15 of Fig. 10.9). One common practice for Python to assign an unknown number of colors is to project the number of clusters

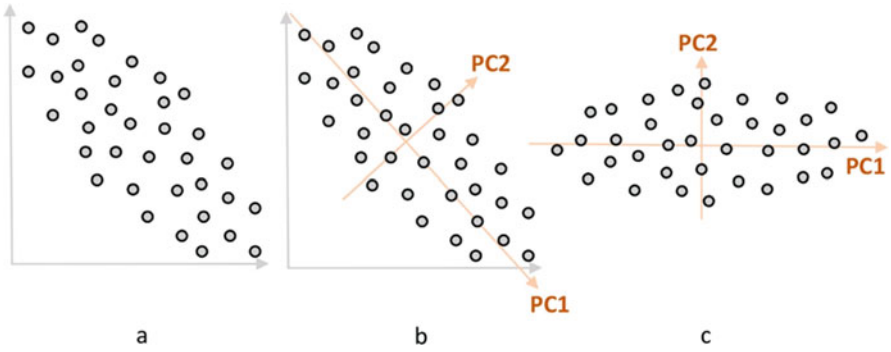


Fig. 10.10 (**a**) A plot of three dozens 2D data points. (**b**) PCA draws the line PC1 such that the data points have the greatest amount of variance along PC1. PCA then draws a second line PC2 that is orthogonal to PC1 and again explains the greatest possible amount of variance along PC2. (**c**) The data points are re-plotted using PC1 and PC2

linearly onto the rainbow colormap (i.e., red, orange, yellow ... blue, and purple.) The three assigned colors in this case are red, green, and purple as shown in lower half of Fig. 10.10.

Bear in mind that the scatterplot in Fig. 10.9 is a 2D scatterplot of EM vs. FI, and the red outliers shown in the same figure are computed using all five parameters (i.e., FI, MU, EM, EV, and S). Red points that are close in proximity in Fig. 10.9 may indeed be far away from each other when all five dimensions are considered. In other words, data features found in a high-dimensional space (e.g., csp.cvs is a 5D dataset) do not always occur in a corresponding low-dimensional setting (e.g., 2D plot in Fig. 10.9). In machine learning, this phenomenon is known as the Curse of Dimensionality (Curse of dimensionality, 2018).

10.3.4 Principal Component Analysis (PCA)

As we mentioned earlier, a primary goal of PCA is to reduce the amount of underlying information, but retain the relevant information necessary to represent the original data. Given a multidimensional dataset, PCA attempts to define a new set of dimensions for the data. Figure 10.10a shows a dataset with dozens 2D data points, PCA finds a straight line (PC1 in Fig. 10.10b) that explains the greatest amount of variance in the data points (or spreading out the most) along that line. We call that line the first principal component and thus PC1 in Fig. 10.10b. Because it is a 2D dataset, PCA can find a second line that is orthogonal to the first one and again maximizes the spreading of the data points. After the two principal axes (PC1 and PC2) are identified, the data points are then projected on the two axes as shown in Fig. 10.10c. The term "projection" means changing the coordinates of the data points based on PC1 and PC2.

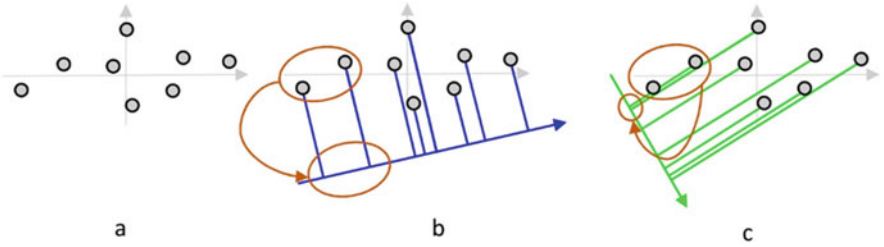


Fig. 10.11 (**a**) Eight 2D data points are plotted. (**b**) The two highlighted points spread out widely in the original 2D spaces. The same two points remain far away when they are projected onto the blue axis. (**c**) The same two highlighted points in the original 2D space almost overlap with each other when they are projected onto the green axis

We want to maximize the variance (or have the greatest spread) because we want to preserve relative distances among the data points. For example, data points in Fig. 10.11a can be projected to the blue axis in Fig. 10.11b or the green axis in Fig. 10.11c. But the projection in Fig. 10.11b better preserves the distance information of the data points (e.g., the two highlighted points) and thus the structure of the original dataset in Fig. 10.11a, whereas the same two highlighted points almost overlap each other in Fig. 10.11c (and fail to preserve the original structure shown in Fig. 10.11a). In other words, the blue axis preserves more variance of the dataset than the green axis.

10.3.4.1 Primary Concept of PCA

The PCA algorithm is rather straightforward, although it requires some linear algebra knowledge to fully appreciate the details. Given a multidimensional dataset X with d dimensions (or variables or attributes) and N instances. The following algorithm treats X as an $N \times d$ matrix.

Centralize the matrix X by subtracting the mean from each attribute so the origin is in the center of the data points.

Compute the $d \times d$ covariance matrix $C = \frac{1}{N-1} X^T X$ such that $C_{i,j} = \frac{1}{N-1} \sum_{k=1}^{N} X_{k,i} . X_{k,j}$ where $X_{k,i} . X_{k,j}$ is the dot product of the two vectors. Given a covariance matrix C, its diagonal $C_{i,i}$ (i.e., the diagonal of the matrix) is the variance of dimension i, whereas $C_{i,j}$ (the off-diagonal of the matrix) is the covariance between dimensions i and j.

Compute the eigenvectors of the above covariance matrix C. The eigenvector with the highest eigenvalue is the first principal component of X, the second highest one is the second principal component and so on.

10.3.4.2 PCA Demo Program Using Python

Figure 10.12 shows the PCA algorithm as implemented using Python 3. Once again, lines 1–3 set up the Python modules required for this program. Lines 5–7 input data from 'cps.csv' file. Lines 9–10 compute the eigenvalues and eigenvectors of the covariance matrix. Lines 12–13 of Fig. 10.12 print out the five normalized eigenvalues in ascending order and then the corresponding five eigenvectors in tabular form in the lower half of Fig. 10.12.

An eigenvalue is indeed the variance along an eigenvector, the first eigenvector alone (in this case) already explains 91.36% of the total variance of the cps.csv dataset. Adding the second vector to the mix will further improve the accuracy of the first two principal components to $91.36\% + 5.69\% \approx 97\%$ of the total variance. Now we can use only two principal components (i.e., 1st and 2nd) to represent the 5D cps.csv dataset in follow-up computation and save up to 60% of memory (as

```
1  import pandas as pd
2  from sklearn import decomposition
3  from sklearn.decomposition import PCA
4
5  cps = pd.read_csv('cps.csv', header=None)
6  cps.columns = ['FI','MU','EN', 'EV', 'S']
7  X = cps.iloc[:,:].values
8
9  pca = decomposition.PCA()
10 pca.fit_transform(X)
11
12 print(pca.explained_variance_ratio_)
13 display(pd.DataFrame(pca.components_, columns=cps.columns))
```

```
[0.91355698 0.05692013 0.01805982 0.00931031 0.00215277]
```

	FI	MU	EN	EV	S
0	0.602496	0.325988	0.723320	0.047994	0.072356
1	0.784025	-0.391908	-0.478342	-0.034100	0.041694
2	0.110188	0.838756	-0.492802	0.048459	0.197850
3	-0.098870	-0.190382	0.056623	0.267136	0.937770
4	-0.019666	0.019567	0.044004	-0.960637	0.272892

Fig. 10.12 A Python program that computes the principal components of the cps.csv file and shows the eigenvalues and the corresponding eigenvectors

we drop 3 out of 5 eigenvectors) and potentially significant amount of computation time, depending on the complexity of the algorithms being applied to the data.

10.3.5 Multidimensional Scaling (MDS)

Given a dataset in R^n, MDS (Cox & Cox, 2001) searches for a low dimensional space (e.g., R^2) in which the 2D points in R^2 represent the corresponding instances of the original dataset in R^n such that the distances between the points in R^2 *approximate* the dissimilarities of the corresponding instances in the original R^n space. Notice that we use the term approximation in the above description, which suggests that precision of the dimension reduction process (i.e., $R^n \rightarrow R^2$) is likely not the most important requirement of the underlying analysis. In fact, MDS is frequently applied to exploratory analysis in data visualization.

There are different strategies to realize the MDS concept. For example, the one we describe below, known as classical scaling (Gower, 1966), uses Euclidean distance to determine the differences (or dissimilarities) between two vectors. Classical scaling belongs to the metric scaling family that also includes least-square scaling techniques such as Sammon mapping (Sammon, 1969). A popular non-metric MDS technique found in the literature is Kruskal's algorithm (Kruskal, 1964).

10.3.5.1 Primary Concept of MDS

The algorithm of classical scaling, which uses Euclidean distance in the input space, is theoretically equivalent (Cox & Cox, 2001) to that of principal component analysis discussed earlier. Classical scaling attempts to preserve Euclidean distances, whereas PCA tries to retain the maximum data variance. Given a multidimensional dataset X with d dimensions and N instances (i.e., an $N \times d$ matrix), PCA solves the $d \times d$ covariance matrix $C = \frac{1}{N-1} X^T X$, whereas MDS solves the $N \times N$ Gram matrix (Gram matrix, 2017) $G = XX^T$. Similar to PCA, the next step for MDS is to solve the eigenvectors of the Gram matrix. For data exploration applications such as data visualization, the first two or three eigenvectors with the highest eigenvalue are often used to plot the N data points in the new MDS space.

Because $N \gg d$ in many big data analytical problems, the computational time required to solve the eigenvector of a big matrix could potentially favor the PCA approach. On the other hand, if only the pairwise distance matrix of the underlying problem is known, MDS is the choice to solve the problem. This happens when, for example, analyzing the rating results by a group of judges in a wine tasting test. In this case, only the pairwise differences among the wines are known.

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3  from sklearn.manifold import MDS
4  from sklearn.metrics import pairwise_distances
5
6  %matplotlib inline
7  plt.figure(figsize=(4, 4))
8
9  cps = pd.read_csv('cps.csv', header=None)
10 X = cps.iloc[:,:].values
11
12 dist=pairwise_distances(X)
13 mds=MDS(n_components=2, dissimilarity='precomputed', random_state=1)
14 pos=mds.fit_transform(dist)
15 xs, ys=pos[:,0], pos[:,1]
16
17 plt.scatter(x=xs, y=ys, s=30, color='green')
18 ax = plt.gca()
19 ax.xaxis.set_visible(False)
20 ax.yaxis.set_visible(False)
21 plt.show()
```

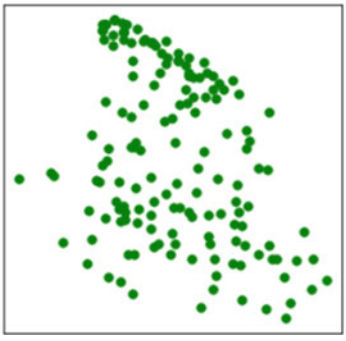


Fig. 10.13 A Python program that computes and plots the two eigenvectors with the largest eigenvalues of the cps.csv dataset

10.3.5.2 MDS Demo Program Using Python

Figure 10.13 shows the MDS program in Python 3. Like all the prior Python program examples in this chapter, lines 1–4 set up Python modules required by the MDS algorithm to run. Lines 6–7 request inline figure output and define the figure sizes (as shown later in Fig. 10.13). Lines 9–10 input data from the 'cps.csv' file. Lines 12–15 solve the eigenvectors of the pairwise Gram matrix and set up the output coordinates of the three first two columns, which are the most important eigenvectors. Lines 17–21 plot the two eigenvectors as a scatterplot as shown in Fig. 10.13.

Figure 10.13 is a layout configuration such that the pairwise Euclidean distances of all the green scatter points in the figure approximate the dissimilarities of the

corresponding 5D data points in the cps.csv dataset. The two axes of the scatterplot in Fig. 10.13 are sometimes referred to as the principal coordinate axes of the configuration. Most of the time, people would simply omit the labels and tic marks of a MDS scatterplot, as we have done in lines 19–20 in Fig. 10.13.

10.4 Discussion and Conclusions

The book chapter discusses the concept and applications of unsupervised machine learning algorithms and techniques in the area of psychometric assessment analytics. Very few data science solutions today are built solely on machine learning algorithms. Even the discussion of this chapter relies on a combination of graphics and databases techniques to supplement the implementation and explanation of the learning processes and results. Indeed, the integration of machine learning and visualization lays the very foundation of the area of visual analytics (Wong & Thomas, 2004). The inclusion of unique database structures beyond a spreadsheet table, such as a data-cube, also plays a critical role in our ongoing research in computational psychometrics.

The discussion is neither comprehensive nor complete. The topic of machine learning will most likely evolve as both computer hardware and software continue to advance. We hope this brief introduction on machine learning can inspire readers to continue exploring the possibility of using machine-learning methods to address their data analytics problems in the future.

References

Anaconda. (2017). *Anaconda*.https://www.anaconda.com/download

Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Communications of the ACM (CACM), 18*(9). ACM Press.

Cox, T. F., & Cox, M. A. A. (2001). *Multidimensional scaling* (2nd ed.). Chapman & Hall/CRC Press.

Curse of Dimensionality. (2018). *Curse of dimensionality – Wikipedia*. https://en.wikipedia.org/wiki/Curse_of_dimensionality

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, 39*(1). Blackwell Publishing.

Domingos, P. (2015). *The master algorithm: How the quest for the ultimate learning machine will remake our world*. Basic Books.

Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of 2nd international conference on knowledge discovery and data mining (KDD-96)* (pp. 227–231).

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI Magazine, 17*(3). AAAI Press.

Feature Extraction. (2017). *Feature extraction – Wikipedia*. https://en.wikipedia.org/wiki/Feature_extraction

Feature Selection. (2017). *Feature selection – Wikipedia*. https://en.wikipedia.org/wiki/Feature_selection

Gower, J. C. (1966). Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika, 53*(3 and 4). Oxford University Press.

Gram Matrix. (2017). *Gramian matrix – Wikipedia*. https://en.wikipedia.org/wiki/Gramian_matrix

Jupyter. (2017). *Project Jupyter | Home*. http://jupyter.org

Kruskal, J. B. (1964). Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika, 29*(1), 1–27. Springer.

Machine Learning. (2017). *Machine learning – Wikipedia*. https://en.wikipedia.org/wiki/Outline_of_machine_learning

Polyak, S. T., von Davier, A., & Peterschmidt, K. (2017). Analyzing game-based collaborative problem solving with computational psychometrics. In *Proceedings ACM KDD 2017 workshop on advancing education with data*. ACM Press.

Python. (2017). *Python – Official site*. https://www.python.org/

R. (2017). *The R project for statistical computing*. https://www.r-project.org/

Ruby. (2017). *Ruby programming language*. https://www.ruby-lang.org/en/

Sammon, J. W. (1969). A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers, 18*(5). IEEE Computer Society Press.

Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development, 3*(3). IBM Press.

Supervised Learning. (2017). *Supervised learning – Wikipedia*. https://en.wikipedia.org/wiki/Supervised_learning

Tukey, J. W. (1997). *Exploratory data analysis*. Pearson.

Unsupervised Machine Learning. (2017). *Unsupervised learning – Wikipedia*. https://en.wikipedia.org/wiki/Unsupervised_learning

Ward's Method. (2017). *Ward's method – Wikipedia*. https://en.wikipedia.org/wiki/Ward%27s_method

Weka. (2017). *Weka – University of Waikato*. http://www.cs.waikato.ac.nz/ml/weka/

Wong, P. C., & Thomas, J. (2004). Visual analytics. *IEEE Computer Graphics and Applications, 24*(5). IEEE Computer Society Press.

Xu, R., & Wunsch, D. C., II (Eds.). (2008). *Clustering*. Wiley/IEEE Press.

Chapter 11
Advances in AI and Machine Learning for Education Research

Yuchi Huang and Saad M. Khan

Abstract There is a growing need for assessment and learning tools that capture a broad range of learner behavior necessary for the evaluation of skills such as problem solving, communication and collaboration. In these real-world applications student data is captured with a high degree of granularity, variety of temporal scales and in a multitude of modalities. Unfortunately such complex, noisy and unstructured data limit the applicability of traditional models of measurement and psychometrics designed to extract evidence of competency from item response data. In this chapter, we present recent advances in AI and Machine Learning that can be utilized for measurement of a variety of complex constructs and competencies. These models include frameworks such as deep neural networks and adversarial generative networks that enable us to harness concept hierarchies and the latent structure within data to learn increasingly complex representations and make powerful predictions.

11.1 Introduction

In traditional psychometrics, digital interactions that were used to make inferences about learners' knowledge, skills, and attributes were constrained due to limitations in the richness and types of data that could be processed automatically. The technology did not exist to process at scale, large amounts of student interaction

The R or Python codes can be found at the GitHub repository of this book: https://github.com/jgbrainstorm/computational_psychometrics

Y. Huang (✉)
ACT, Iowa City, IA, USA
e-mail: yuchi.huang@act.org

S. M. Khan
FineTune Learning, New York, NY, USA
e-mail: saad@finetunelearning.com

© Springer Nature Switzerland AG 2021

A. A. von Davier et al. (eds.), *Computational Psychometrics: New Methodologies for a New Generation of Digital Learning and Assessment*, Methodology of Educational Measurement and Assessment, https://doi.org/10.1007/978-3-030-74394-9_11

data and extract particular features from student performance beyond responses to highly structured item responses in an automated fashion. Nevertheless, there is a growing need for assessment and learning tools that capture a broad range of learner behavior necessary for the evaluation of skills such as problem solving (Hao et al., 2017; Polyak et al., 2017), communication and collaboration (von Davier et al., 2017). A key feature of such tools is the use of digital interfaces that enable rich, immersive interactions and can capture student data in a multitude of sensory modalities and accessible formats in real world activities. But how do we extract meaningful evidence of construct competency from such unstructured data captured in vivo and how do we utilize big-data analytical and computational methods to expand the scale and scope of traditional psychometric areas of enquiry and modeling?

Advances in fields such as machine learning, educational data mining, computer vision and natural language processing are making it possible to analyze and score a much wider range of responses. This is enabling the expansion of the types of activities that can be used to characterize the knowledge, skills and attributes of learners and, hence, an expansion of the forms of assessment we can use. Building on traditional psychometrics, computational psychometrics has incorporated techniques from above fields to analyze large-scale/high-dimensional learning, assessment, biometric, or psychological data and provided actionable and meaningful feedback to individuals based on measurement of personal differences as they pertain to specific skills and areas of enquiry (von Davier, 2018). These competencies may include a wide variety of cognitive and non-cognitive skills, twenty-first century competencies like communication, collaboration, problem solving (e.g., diagnosing patients), troubleshooting (e.g., computer networks etc.) and related workforce readiness skills.

The analysis of such skills entails the recognition and extraction of patterns of behaviors and activities from the time-series data that constitutes interaction between individuals as well as with the task. This data is very rich and encompasses multiple modalities i.e., in addition to traditional forms of computer interaction data like mouse clicks, keystrokes etc., this data includes audio and video of humans interacting with each other and the simulation task. Inexpensive and ubiquitous sensors like webcams and microphones enable comprehensive sensing of the user's behavior with real-time capture and recognition of the users's gaze, facial expressions, gestures, body pose, spoken responses and speech tone. These different modalities can then be integrated to model the user's cognitive thought processes and non-cognitive behavior like engagement, motivation, persistence and affective (emotional) state.

11.2 Deep Neural Networks to Implicitly Model Inherent Deep Structures in Complex Data

In Chap. 3, computational psychometrics is defined to utilize advanced measurement techniques that combine models from different domains such as machine learning, computer vision, educational data mining and natural language processing to analyze large-scale/high-dimensional data. In the following, we provide some details on these approaches. In particular a sub-field of machine learning, deep learning, has resulted in new computational architectures that are well suited to extracting meaningful semantic features from highly complex and noisy multimodal data.

Deep learning is a statistical technique for extracting patterns by using neural networks with multiple layers. It is part of a large family of machine learning methods based on learning implicit data representations. Deep learning can be supervised, semi-supervised or unsupervised. Its predecessor is known as artificial neural Network (ANN) historically, which only consists of 2 ~ 3 layers of connected units or nodes called artificial neurons which loosely model the neurons in animal brains. Unfortunately, ANN didn't scale to larger problems and was moved back into storage due to limits of data and computing power; by the 90s the support vector machine (SVM) became the method of choice.

Around 2006, Hinton rebranded ANN as 'deep learning' by introducing the idea of unsupervised pre-training and deep belief nets (Hinton, 2007), in which people could train networks that were deeper than previous attempts. Breakthrough happened in 2012, with the publication of Alex Krizhevsky, Ilya Sutskever, and Geoff Hinton's highly influential work 'ImageNet Classification with Deep Convolutional Neural Networks' (Krizhevsky et al., 2017), which achieved state-of-the-art results on the object recognition challenge known as ImageNet by halving the previous best error rate. The most important contribution of this work is the use of massive labeled data sets and graphics processing units (GPUs) for model training. Since then, deep learning architectures such as deep neural networks, deep convolutional networks and recurrent neural networks have yielded numerous state of the art results, in domains such as speech recognition, image recognition, and language translation and plays a role in a wide swath of current AI applications. In some cases, they have produced results comparable to or even superior to human experts. Its most attractive quality is that it can perform well without external human-designed resources or time-intensive feature engineering. Despite the advantage there has been limited use of deep networks in educational applications. Some very recent work has emerged such as using deep learning models to predict student learning outcomes and grades from granular interaction data (Kim et al., 2018) as well as predicting student dropouts in MOOCs (Dalipi et al., 2018). We believe there are many applications that remain unexplored and highly suitable for deep neural networks such as educational data classification with complex standards and taxonomies, time series data modeling and new content generation.

The purpose of this chapter is to briefly introduce the inner working mechanism of deep learning techniques and inspire their applications in computational psychometrics and other Educational domains. Section 11.3 presents static models to build robust classifiers and evidence detectors of construct competency: Deep Neural Networks (DNNs) (Bengio, 2009) and Convolutional Neural Networks (CNNs) (LeCun & Bengio, 1995). Sect. 11.4 covers Recurrent Neural Networks (RNN) (Miljanovic, 2012), a temporal deep model to capture interaction dynamics and time-series process data, as a useful complement to the methods in Chap. 10. Long short-term memory (LSTM) (Hochreiter & Schmidhuber, 1997) units (or blocks) will be introduced as the building blocks for layers of RNN. In Sect. 11.5, we will discuss how generative deep models such as Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) could be used in fully autonomous interaction between learner/test-taker/user and AI agent. By this point, the readers should have a rough understanding on the advantages of various deep learning systems and their limitations.

11.3 Static Models to Build Robust Classifiers and Evidence Detectors: DNNs and CNNs

The term Deep Neural Networks (DNN) refers artificial neural network (ANN) with multiple hidden layers between the input and output layers, which are typically feedforward networks in which data flows from the input layer to the output layer without looping back. DNN frameworks can model complex non-linear relationships. They are designed to recognize patterns into which all real-world data such as images, sound, text or time series data, must be translated.

A key component of DNNs (or ANNs) is the single-hidden-layer Multi-Layer Perceptron (MLP) (Rosenblatt, 1961). An MLP can be viewed as a logistic regression classifier where the input is first transformed using a learnt non-linear transformation. This transformation projects the input data into a space where it becomes linearly separable. An MLP with a single hidden layer can be represented mathematically as follows:

$$f(x) = G\left(b^{(2)} + W^{(2)}\left(s\left(b^{(1)} + W^{(1)}x\right)\right)\right),$$

where $b^{(1)}$ and $b^{(2)}$ are bias vectors; $W^{(1)}$ and $W^{(2)}$ are weight matrices; G and s are nonlinear activation functions (such as tanh, sigmoid or rectified linear).

As shown in Fig. 11.1, the intermediate layer is referred to as a hidden layer. A single hidden layer or the composition of 2 ~ 3 layers of MLP is sufficient to act as a universal approximator. However, there are substantial benefits to using many such hidden layers together to form DNNs: the extra layers of DNNs enable abstraction of features, potentially modeling complex data with fewer units than a similarly performing shallow network. The number of layers they could contain are the major

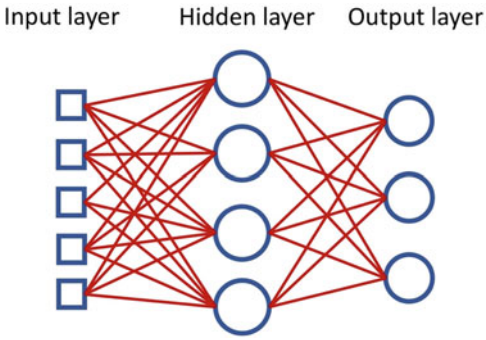


Fig. 11.1 Illustration of a single-hidden-layer Multi-Layer Perceptron

difference between ANNs and DNNs. Backpropagation (Rumelhart et al., 1986) is used to learn the parameters in all layers; this technique is also sometimes called backward propagation of errors, because the error is calculated at the output layer and distributed back through all the network layers.

Although fully connected DNNs can be used to learn features as well as classify data, it is not practical to apply them to high-dimensional data such as images. A large number of neurons would be used due to the very large input sizes associated with images, where each pixel is a relevant variable. For instance, a fully connected layer for a small image of size 100×100 has 10,000 weights for each neuron in the second layer.

In this way, DNNs are prone to overfitting because of a large number of free parameters that added layers introduced, which allow them to model rare dependencies (e.g. noise) in the training data. Regularization methods such as weight decay (Krogh & Hertz, 1992) can be applied during training to combat overfitting. Dropout technique could also be applied to randomly omits units from the hidden layers during training. This helps to restrain overfitting (Hinton et al., 2012). Finally, data can be augmented via methods such as cropping and rotating such that smaller training sets can be increased in size to reduce the chances of overfitting.

Alternatively, engineers have looked for other types of neural networks with more straightforward structures and convergent training algorithms. Convolutional neural networks (CNN) are specific type of deep neural networks which are biologically-inspired variants of MLPs. Cells in the visual cortex are sensitive to small sub-regions of the visual field, which are tiled to cover the entire visual field. These cells act as local filters over the input space and are well-suited to exploit the strong spatially local correlation present in input signals. The basic components of CNNs are convolutional layers which imitates the functions of visual cortex. The layer's parameters consist of a set of learnable kernels, which have a small receptive field, but extend through the full depth of the input volume. During the forward pass, each kernel slides through the width and height of the input volume, computes the dot product between the entries of the filter and the input to produce a 2-dimensional feature map with respect to that kernel. Similar to DNNs, the

kernels in convolutional layers along with parameters in other layers of CNNs (such as pooling and full connection layers) are learned by Backpropagation. The architecture of the famous 'AlexNet' (Krizhevsky et al., 2017), which made the first breakthrough of deep learning on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012, consisted of 8 layers of neural networks containing a total of nearly 60 million parameters. The winner of the later ILSVRC 2014 competition was 'GoogleNet' (Szegedy et al., 2015) that consisted a much deeper 22 layer CNN. In ILSVRC 2015, a novel architecture called Residual Neural Network (ResNet) (He et al., 2015) was introduced with "skip connections" and heavy batch normalization. This framework contains 152 layers and for the first time beat human-level performance on the ImageNet dataset.

CNNs models are especially useful in various sub-domains of image and video recognition, recommender systems and natural language processing. For example, CNNs could be applied to detect and analyze the dynamics in classrooms, such as movements of teachers and students and interactions between them (Hammerla et al., 2016), their emotions (Levi & Hassner, 2015) etc.; CNNs could also be utilized in video analysis and recommendation for Online Educational Platform (Feichtenhofer et al., 2016); they could also be used on educational text data to perform semantic parsing, search query retrieval, sentence modeling (Kalchbrenner et al., 2014), etc. In Fig. 11.2, we provide the Python code to measure distance of two images by applying a popular CNN architecture – VGG (Simonyan & Zisserman, 2014). In this code, we utilized a pre-trained VGG model of the famous machine learning package Pytorch and performed the inference on two images using the VGG model. The inferenced features of the last hidden layer can be employed to measure the semantic distance of two images in the feature space. This code could be widely used in object recognition, image classification and searching.

11.4 Deep Networks to Model Time Series Process Data: RNNs

In a traditional neural network, we assume that all inputs (and outputs) are independent of each other; but for many tasks that is not the case. If we want to predict the next word in a sentence, we better know which words came before it. *Recurrent* neural networks (RNNs) (Miljanovic, 2012) are artificial neural networks designed to recognize patterns in sequential data, such as text, genomes, handwriting, the spoken word, or numerical times series data emanating from sensors, stock markets and government agencies. These algorithms have a temporal dimension by taking time and sequence into account. RNNs possess a certain type of memory and memory is also part of the human condition, that is why repeated analogies have been made between memory of brain and that of RNNs. RNNs are proven be one of the most powerful and useful type of neural networks, alongside the memory networks and attention mechanism. The purpose of this part is to give readers an intuition about the functioning of RNNs and the structure of two prominent RNN

Load the needed libraries

```
import torch
import torch.nn as nn
import torchvision.models as models
from torchvision import transforms
from torch.autograd import Variable
from PIL import Image
import numpy as np
```

Load the vgg model

```
vgg19 = models.vgg19(pretrained=True)
modules=list(vgg19.children())[:-1]
vgg19 = nn.Sequential(*modules)

for p in vgg19.parameters():
    p.requires_grad = False
```

Preprocessing

```
#Define the normalization transform to normalize the image int
ensities
normalize = transforms.Normalize(
   mean=[0.485, 0.456, 0.406],
   std=[0.229, 0.224, 0.225])

#Define the preprocessing transform to perform the scaling of
the images
preprocess = transforms.Compose([
   transforms.Scale((224,224)),
   transforms.CenterCrop(224),
   transforms.ToTensor(),
   normalize])
```

Feature extracting and distance computation

```
# import and extract a feature from image1
img_pil_first = Image.open("first.png")
img_tensor_first = preprocess(img_pil_first)
img_tensor_first.unsqueeze_(0)
img_var_first = Variable(img_tensor_first) # assign it to a va
riable
features_var_first = vgg19(img_var_first) # get the output fro
m the last hidden layer of the pretrained model
features_first = features_var_first.data # get the tensor out
of the variable
feat_first = torch.squeeze(features_first)
feat_first = feat_first.numpy()
#print(feat_first)

# import and extract a feature from image2
img_pil_first = Image.open("second.png")
img_tensor_last = preprocess(img_pil_last)
img_tensor_last.unsqueeze_(0)
img_var_last = Variable(img_tensor_last)
features_var_last = vgg19(img_var_last)
features_last = features_var_last.data # get the tensor out of
the variable
feat_last = torch.squeeze(features_last)
feat_last = feat_last.numpy()
print("the distance between two images")
print(np.sqrt(np.sum(np.power(feat_first - feat_last, 2))))
```

Fig. 11.2 Python codes for measuring the distance of two images

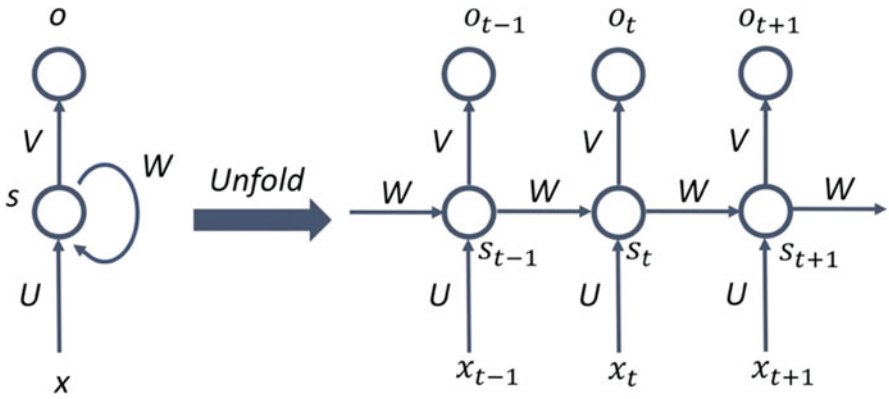


Fig. 11.3 A simple RNN and its unfolded version

variations, Long Short Term Memory unit (LSTM) (Hochreiter & Schmidhuber, 1997) and Gated recurrent unit (GRU) (Cho et al., 2014).

Different from feedforward networks such as Deep Neural Networks, RNN takes as their input not just the current input example they see, but also what they have perceived previously in time. They perform the same task for every element of a sequence, with the output being depended on the previous computations. Another way to think about RNN is that they have a "memory" which captures information about what has been calculated so far. In theory RNN can make use of information in arbitrarily long sequences, but in practice they are limited to looking back only a few steps. Basic RNN are a network of neuron-like nodes, each with a one-way connection to every other node. Each node has a time-varying real-valued activation. Each connection has a modifiable real-valued weight. Nodes are either input nodes, output nodes, or hidden nodes that modify the data on route from input to output. Figure 11.3 illustrates a typical RNN and its unfolded version. By unfolding we mean that we write out the network for the complete sequence. The formulas that govern the computation happening in an RNN are as follows:

- x_t is the input at time step t. For instance, x_0 could be a one-hot vector corresponding to the first word of a sentence.
- s_t is the hidden state at time step t, which contains the "memory" of the network and captures information about what happened in all the previous time steps. In practice s_t typically can't capture information from too many time steps ago. s_t is calculated based on the previous hidden state and the input at the current step: $s = f(Ux_t + Ws_{t-1})$. The function f is a nonlinear activation such as *tanh* or ReLU (Nair & Hinton, 2010). RNN shares the same parameters (U, V, W) across all steps, which reduces the total number of parameters we need to learn. s_{-1} is typically initialized to all zeroes, which is required to calculate the first hidden state.

- o_t is the output at step t. For example, in order to predict the next word in a sentence, we need to produce a vector of probabilities across our vocabulary. $o_t = softmax(Vs_t)$. Depending on the task outputting at each time step may not be necessary. For example, when predicting the sentiment of a sentence, we may only need about the final output.

Applications of Recurrent Neural Networks include robot control, time series prediction, speech recognition, grammar learning, human action recognition, business process management, and prediction in medical care pathways among others. RNNs are especially useful in various problems in natural language processing (NLP), such as semantic parsing and question answering (Tan et al., 2015), document summarization (Paulus et al., 2017), machine translation (Vaswani et al., 2018) etc., which are all important tasks in Education.

11.5 Generative Models to Learn Real World Data Distribution: GANs

A Generative Model is a method to learn true data distribution of real world from training set then to generate new data points with some variations. Generative models are one of the most promising approaches towards the goal of understanding the world. The intuition behind this approach follows a famous quote from Richard Feynman "What I cannot create, I do not understand."

Generative adversarial networks (GANs) (Goodfellow et al., 2014) have dramatically sharpened the possibility of AI-generated content and have drawn active research efforts since they were first described by Ian Goodfellow et al. in 2014. It is implemented by a system of two neural networks (Generator Neural Network and Discriminator Neural Network) contesting with each other in a zero-sum game framework (see Fig. 11.4). The generative network $G(z)$ learns to map random

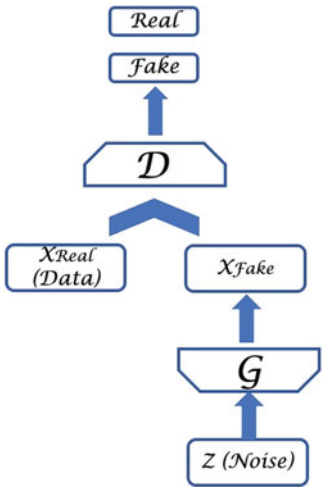


Fig. 11.4 Basic architecture of a GAN model

inputs z from a latent space to a particular data distribution of interest, while the discriminative network $D(x)$ discriminates between instances from the true data distribution and candidates produced by the generator. The generative network's training objective is to increase the error rate of the discriminative network ("fool" the discriminator network by producing fake synthesized instances that appear to have come from the true data distribution). The task of Discriminator Network is to take input x either from the real data or from the generator $G(z)$ and try to predict whether the input is real or generated (1 or 0). This can be represented mathematically as the following equation:

$$\min_G \max_D \left(\mathbb{E}_{x \sim p_{data}(x)} [log D(x)] + \mathbb{E}_{z \sim p_z(z)} \left[\log (1 - D(G(z))) \right] \right)$$

the first term of this Equation is entropy that the data from real distribution passes through the discriminator and the discriminator tries to maximize $D(x)$ to 1. The second term is entropy that the data from random noise passes through the generator, which generates a fake sample passing through the discriminator. In this term, discriminator tries to minimize $D(G(z))$ to 0. However, the other hand, the task of generator is exactly opposite, i.e., tries to maximize $D(G(z))$ to 1.

Previous work based on GANs has shown promise in a number of applications such as future state prediction of time series data (Mathieu et al., 2016), video generation (Vondrick et al., 2016), image manipulation (Zhu et al., 2016), style transfer (Li & Wand, 2016), text-to-image (Isola et al., 2017)/image-to-image translation (Reed et al., 2016) and 3D shape modeling (Wu et al., 2016). In (Huang & Khan, 2018a, b), human dyadic collaborative interactions are studied to tackle the problem of generating video sequences of dynamically varying facial expression in human-agent interactions using conditional GANs. Different from previous work, they do not employ conditions related to facial attributes of generated agent identities but consider the influence of the interacting human's facial expressions in the virtual agent response.

To give readers a concrete sense of how the GAN model used in practice, in Fig. 11.5, we provide the Python example functions on generating facial expression images from sketch images (Huang & Khan, 2018a, b) based on the Tensorflow package (Abadi et al., 2015). In the example code, we only show the discriminator and the generator functions due to the space limit; it also contains some utility functions such as lrelu (leaky rectified linear unit function), conv2d (2d convolution), deconv2d (2-d de-convolution) etc. This code could be widely used in different image generation procedure.

The sample code of the discriminator and the generator of a GAN model

```
import tensorflow as tf
import numpy as np

def discriminator(self, image, y=None, reuse=False):
    with tf.variable_scope("discriminator") as scope:
        if reuse:
            tf.get_variable_scope().reuse_variables()
        else:
            assert tf.get_variable_scope().reuse == False

        h0 = lrelu(conv2d(image, self.df_dim, name='d_h0_conv'))
        h1 = lrelu(self.d_bn1(conv2d(h0, self.df_dim*2, name='d_h1_conv')))
        h2 = lrelu(self.d_bn2(conv2d(h1, self.df_dim*4, name='d_h2_conv')))
        h3 = lrelu(self.d_bn3(conv2d(h2, self.df_dim*8, d_h=1, d_w=1, name='d_h3_co
nv')))
        h4 = linear(tf.reshape(h3, [self.batch_size, -1]), 1, 'd_h3_lin')
        return tf.nn.sigmoid(h4), h4, h3, h2, h1, h0

def generator(self, image, y=None):
    with tf.variable_scope("generator") as scope:
        s = self.output_size
        s2, s4, s8, s16, s32, s64, s128 = int(s/2), int(s/4), int(s/8), int(s/16),
int(s/32), int(s/64), int(s/128)

        e1 = conv2d(image, self.gf_dim, name='g_e1_conv')
        e2 = self.g_bn_e2(conv2d(lrelu(e1), self.gf_dim*2, name='g_e2_conv'))
        e3 = self.g_bn_e3(conv2d(lrelu(e2), self.gf_dim*4, name='g_e3_conv'))
        e4 = self.g_bn_e4(conv2d(lrelu(e3), self.gf_dim*8, name='g_e4_conv'))
        e5 = self.g_bn_e5(conv2d(lrelu(e4), self.gf_dim*8, name='g_e5_conv'))
        e6 = self.g_bn_e6(conv2d(lrelu(e5), self.gf_dim*8, name='g_e6_conv'))
        e7 = self.g_bn_e7(conv2d(lrelu(e6), self.gf_dim*8, name='g_e7_conv'))
        e8 = self.g_bn_e8(conv2d(lrelu(e7), self.gf_dim*8, name='g_e8_conv'))

        self.d1, self.d1_w, self.d1_b = deconv2d(tf.nn.relu(e8),
            [self.batch_size, s128, s128, self.gf_dim*8], name='g_d1', with_w=True)
        d1 = tf.nn.dropout(self.g_bn_d1(self.d1), 0.5)
        d1 = tf.concat_v2([d1, e7], 3)
        self.d2, self.d2_w, self.d2_b = deconv2d(tf.nn.relu(d1),
            [self.batch_size, s64, s64, self.gf_dim*8], name='g_d2', with_w=True)
        d2 = tf.nn.dropout(self.g_bn_d2(self.d2), 0.5)
        d2 = tf.concat_v2([d2, e6], 3)
        self.d3, self.d3_w, self.d3_b = deconv2d(tf.nn.relu(d2),
            [self.batch_size, s32, s32, self.gf_dim*8], name='g_d3', with_w=True)
        d3 = tf.nn.dropout(self.g_bn_d3(self.d3), 0.5)
        d3 = tf.concat_v2([d3, e5], 3)
        self.d4, self.d4_w, self.d4_b = deconv2d(tf.nn.relu(d3),
            [self.batch_size, s16, s16, self.gf_dim*8], name='g_d4', with_w=True)
        d4 = self.g_bn_d4(self.d4)
        d4 = tf.concat_v2([d4, e4], 3)
        self.d5, self.d5_w, self.d5_b = deconv2d(tf.nn.relu(d4),
            [self.batch_size, s8, s8, self.gf_dim*4], name='g_d5', with_w=True)
        d5 = self.g_bn_d5(self.d5)
        d5 = tf.concat_v2([d5, e3], 3)
        self.d6, self.d6_w, self.d6_b = deconv2d(tf.nn.relu(d5),
            [self.batch_size, s4, s4, self.gf_dim*2], name='g_d6', with_w=True)
        d6 = self.g_bn_d6(self.d6)
        d6 = tf.concat_v2([d6, e2], 3)
        self.d7, self.d7_w, self.d7_b = deconv2d(tf.nn.relu(d6),
            [self.batch_size, s2, s2, self.gf_dim], name='g_d7', with_w=True)
        d7 = self.g_bn_d7(self.d7)
        d7 = tf.concat_v2([d7, e1], 3)
        self.d8, self.d8_w, self.d8_b = deconv2d(tf.nn.relu(d7),
            [self.batch_size, s, s, self.output_c_dim], name='g_d8', with_w=True)
        return tf.nn.tanh(self.d8)
```

Fig. 11.5 Python codes for using GAN

11.6 Summary

A major challenge in the design of educational learning and assessment systems is the ability to create effective computational models that can learn representations of highly complex data. Advances in AI and deep learning have provided a promising solution to this challenge. They allow us to develop intermediate representations of data and fuse information at vastly different levels of abstraction and distil actionable insights on learner's cognitive thought processes and social emotional behavior. In this chapter we have introduced some of the most popular deep learning architectures including CNN, RNN and GAN. These models can be used in a plethora of educational technology applications including analysis of multimodal data such as detection and generation of facial expressions in dyadic collaborative interactions among others.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., . . . , & Zheng, X. (2015). *TensorFlow: Large-scale machine learning on heterogeneous systems*.

Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends® in Machine Learning, 2*(1), 1–127. https://doi.org/10.1561/2200000006

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078.

Dalipi, F., Shariq, A., & Kastrati, Z. (2018). MOOC dropout prediction using machine learning techniques: Review and research challenges. In *9th IEEE global engineering education conference (EDUCON 2018)*.

Feichtenhofer, C., Pinz, A., & Zisserman, A. (2016). Convolutional two-stream network fusion for video action recognition. *CoRR*, abs/1604.06573.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. C., & Bengio, Y. (2014). Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, & K. Q. Weinberger (Eds.), *NIPS* (pp. 2672–2680).

Hammerla, N. Y., Halloran, S., & Ploetz, T. (2016). Deep, convolutional, and recurrent models for human activity recognition using wearables. *CoRR*, abs/1604.08880.

Hao, J., Chen, L., Flor, M., Liu, L., & von Davier, A. A. (2017). CPS-rater: Automated sequential annotation for conversations in collaborative problem-solving activities. *ETS Research Report Series*, 1–9.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. *CoRR*, abs/1512.03385.

Hinton, G. E. (2007). Learning multiple layers of representation. *Trends in Cognitive Sciences, 11*, 428–434.

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*, 1735–1780.

Huang, Y., & Khan, S. M. (2018a). Generating photorealistic facial expressions in dyadic interactions. *BMVC*, 201. BMVA Press.

Huang, Y., & Khan, S. M. (2018b). A generative approach for dynamically varying photorealistic facial expressions in human-agent interactions. In S. K. D'Mello, P. G. Georgiou, S. Scherer, E. M. Provost, M. Soleymani, & M. Worsley (Eds.), *ICMI* (pp. 437–445). ACM.

Isola, P., Zhu, J.-Y., Zhou, T., & Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In *CVPR* (pp. 5967–5976). IEEE Computer Society. ISBN:978-1-5386-0457-1.

Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A convolutional neural network for modelling sentences. *CoRR*, abs/1404.2188.

Kim, B., Vizitei, E., & Ganapathi, V. (2018). GritNet: Student performance prediction with deep learning. In *11th international conference on educational data mining (EDM 2018)* (pp. 625–629).

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM, 60*(6), 84–90. https://doi.org/10.1145/3065386

Krogh, A., & Hertz, J. A. (1992). A simple weight decay can improve generalization. In D. S. Lippman, J. E. Moody, & D. S. Touretzky (Eds.), *Advances in neural information processing systems 4* (pp. 950–957). Morgan Kaufmann.

Lecun, Y., & Bengio, Y. (1995). Convolutional networks for images, speech, and time-series. In M. A. Arbib (Ed.), *The handbook of brain theory and neural networks*. MIT Press.

Levi, G., & Hassner, T. (2015). Emotion recognition in the wild via convolutional neural networks and mapped binary patterns. In Z. Zhang, P. Cohen, D. Bohus, R. Horaud, & H. Meng (Eds.), *ICMI* (pp. 503–510). ACM. ISBN:978-1-4503-3912-4.

Li, C., & Wand, M. (2016). Precomputed real-time texture synthesis with Markovian generative adversarial networks. *CoRR*, abs/1604.04382.

Mathieu, M., Couprie, C., & LeCun, Y. (2016). Deep multi-scale video prediction beyond mean square error. In Y. Bengio & Y. LeCun (Eds.), *ICLR*.

Miljanovic, M. (2012). Comparative analysis of recurrent and finite impulse response neural networks in time series prediction. *Indian Journal of Computer and Engineering*.

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted Boltzmann machines. In J. Fürnkranz & T. Joachims (Eds.), *ICML* (pp. 807–814). Omnipress.

Paulus, R., Xiong, C., & Socher, R. (2017). A deep reinforced model for abstractive summarization. *CoRR*, abs/1705.04304.

Polyak, S. T., von Davier, A. A., & Peterschmidt, K. (2017). Computational psychometrics for the measurement of collaborative problem solving skills. *Frontiers in Psychology, 8*, 20–29. https://doi.org/10.3389/fpsyg.2017.02029

Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., & Lee, H. (2016). Generative adversarial text to image synthesis. *arXiv preprint arXiv*, 1605.05396.

Rosenblatt, F. (1961). Principles of neurodynamics: Perceptrons and the theory of brain mechanisms. In *Spartan books*.

Rumelhart, D. E., Hinton, G. E., & Wilson, R. J. (1986). Learning representations by back-propagating errors. *Nature, 323*, 533–536.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1–9).

Tan, M., Xiang, B., & Zhou, B. (2015). LSTM-based deep learning models for non-factoid answer selection. *CoRR*, abs/1511.04108.

Vaswani, A., Bengio, S., Brevdo, E., Chollet, F., Gomez, A. N., Gouws, S., Jones, L., Kaiser, L., Kalchbrenner, N., Parmar, N., Sepassi, R., Shazeer, N., & Uszkoreit, J. (2018). Tensor2Tensor for neural machine translation. *CoRR*, abs/1803.07416.

Vondrick, C., Pirsiavash, H., & Torralba, A. (2016). Generating videos with scene dynamics. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, & R. Garnett (Eds.), *NIPS* (pp. 613–621).

von Davier, A. A., Hao, J., Liu, L., & Kyllonen, P. (2017). Interdisciplinary research agenda in support of assessment of collaborative problem solving: Lessons learned from developing a collaborative science assessment prototype. *Computers in Human Behavior, 76*, 2017.

von Davier, A. A. (2018). Computational psychometrics in support of collaborative educational assessments. *Journal of Educational Measurement, 54*(1), 3–11. https://doi.org/10.1111/jedm.12129

Wu, J., Zhang, C., Xue, T., Freeman, B., & Tenenbaum, J. (2016). Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, & R. Garnett (Eds.), *NIPS* (pp. 82–90).

Zhu, J.-Y., Krähenbühl, P., Shechtman, E., & Efros, A. A. (2016). Generative visual manipulation on the natural image manifold. *CoRR*, abs/1609.03552.

Chapter 12
Time Series and Stochastic Processes

Peter Halpin, Lu Ou, and Michelle LaMar

Abstract This chapter addresses some statistical modeling approaches for time series data and discusses their potential for psychometric applications. We adopt a broad conceptualization of time series, including under this rubric any type of data that involves serial statistical dependence. Such dependence may be represented in continuous time, discrete time, or in a purely sequential manner. This chapter begins by discussing the relationships among these three representations and offers some general advice on when each might prove useful. We then provide an overview of three modeling frameworks that exemplify the different representations of statistical dependence: Markov decision processes, state-space modeling, and temporal point processes. For each modeling framework, we discuss its specification, its psychometric interpretation, and provide a brief numeric example including `R` code.

12.1 Introduction

Computer-based assessments are increasingly being designed to record finely-grained time series data on respondents' task-related activities (Ercikan & Pellegrino 2017). These data can provide important insights about respondents' cognitive, behavioral, and emotional processes during the completion of an assess-

The R code for this chapter can be found at the GitHub repository of this book: https://github.com/jgbrainstorm/computational_psychometrics.

P. Halpin (✉)
University of North Carolina at Chapel Hill, Chapel Hill, NC, USA
e-mail: peter.halpin@unc.edu

L. Ou
Unaffiliated, Campbell, CA, USA

M. LaMar
Educational Testing Service, Princeton, NJ, USA
e-mail: Lu.Ou@act.org

© Springer Nature Switzerland AG 2021

A. A. von Davier et al. (eds.), *Computational Psychometrics: New Methodologies for a New Generation of Digital Learning and Assessment*, Methodology of Educational Measurement and Assessment, https://doi.org/10.1007/978-3-030-74394-9_12

ment, and consequently have been termed *process data* in the psychometric literature. While process data present transformative opportunities to re-imagine educational and psychological testing, they are not straightforward to model from a psychometric perspective. In particular, process data are incompatible with foundational psychometric assumptions about local independence (Lord & Novick 1968), which require that response data do not demonstrate statistical dependence after conditioning on fixed traits of the respondent. Instead, process data often possess an inherently sequential quality that is more naturally considered from a time series perspective—i.e., by conditioning on past events rather than static traits. To address this issue, this chapter considers how statistical methods for analyzing time series data can be applied in psychometric settings.

We take a broad perspective time series data, and suggest that there are three main ways of modeling this type of data. For heuristic purposes, let us introduce a sequence of random variables $(Y_n)_{n\in\mathbb{N}}$ and consider what the indices n are intended to represent. In a *purely sequential* model, the indices represent only the order of the variables in the sequence. This type of representation is in keeping with the conventional mathematical definition of a sequence, and is exemplified by the Markov decision process discussed in Sect. 12.2. In a *discrete time* model, the sequence of indices also represent equal units of time—i.e., the duration of time elapsed between Y_i and Y_{i+n} is assumed to be equal to that between Y_j and Y_{j+n} for all $i, j, n \in \mathbb{N}$. While equidistant time intervals are not always made explicit in time series models, they are often implicit in the interpretation of central concepts such as lag and autoregression. One way to deal with violations of this assumption is through *continuous time* models, which explicitly incorporate the time duration between subsequent measurement occasions. The state-space model presented in Sect. 12.3 shows how discrete-time models can be extended to continuous time in this manner. The fourth section discusses another continuous time approach, temporal point processes, in which the event times themselves are the focal random variables to be modeled.

Table 12.1 summarizes these three modeling approaches with respect to some other key attributes. The following sections provide brief but self-contained overviews of these three modeling frameworks, emphasizing the model specification and its interpretation in psychometric settings, and providing a minimally reproducible example in version 4 of the `R` language. Additional materials for the examples are provided with the online resources for this book.

This short chapter does not attempt to provide anything close to a comprehensive review of literature on time series and stochastic processes. Rather, our approach is to exemplify each of the three main ways of modeling time using a specific example. In the concluding section, we summarizing the limitations of these modeling approaches and make recommendations for future research on the psychometric analysis of process data.

Table 12.1 Comparison of models to be discussed

Model	Time	Lag	Response variable	Covariates
Markov decision processes	Sequential	Markov	Discrete	None
State-space models	Discrete or continuous	User-specified	Continuous	Time-varying and -constant
Point processes	Continuous	Full history	Time-to-event	Time-varying and -constant

12.2 Markov Decision Processes

In complex task environments, performances often involve a sequence of context-dependent decisions. When the outcomes of these decisions are non-deterministic, the decision maker has to consider possible consequences of each action and how the chosen action might better position them to achieve their goals. Examples of such tasks include experimental inquiry and problem solving, stochastic games, and tasks that involve interacting with another human such as negotiation, collaboration, or competition. To understand how individuals differ in their decision making in such environments, we employ a model for sequential decision making in the presence of uncertainty, the Markov decision process (MDP).

12.2.1 Modeling Framework

A (MDP) is a model for sequential decision making in the presence of uncertainty based on a longitudinal cost-benefit analysis (Puterman 1994). MDPs have been used extensively in artificial intelligence and robotics to choose optimal actions in stochastic situations (Russell & Norvig 2009; Mnih et al. 2015), and in economics to model individual choice strategies (Rust 1994).

Mathematically, an MDP is a model in which a sequence of values of a set K of variables $X_t = \{X_{0t}, X_{1t}, \ldots, X_{tK}\}$, collectively referred to as the system state, are modeled at time t as depending on their values at time $t-1$ as well as an action y taken at time $t-1$. In turn, actions taken are dependent upon the current system state and a reward structure that defines values for particular states.

$$p(y_t|X_t) = \pi(y_t, X_t, R), \tag{12.1}$$

$$p(X_t|y_{t-1}, X_{t-1}) = T(X_{t-1}, y_{t-1}) \tag{12.2}$$

Thus actions are chosen according to the policy, π, and state variables change according to the transition function T.

In most applications of MDPs, the policy is taken as a function that maximizes, in some form, the rewards produced by the states reached. The Q function quantifies the total expected discounted reward for taking a particular action given a particular system state as,

$$Q(X_t, y_t) = \sum_{X_{t+1}} p(X_{t+1}|X_t, y_t)\left(R(X_{t+1}) + \gamma \sum_{y_{t+1}} p(y_{t+1}|X_{t+1})Q(X_{t+1}, y_{t+1})\right). \tag{12.3}$$

The term $R(X_{t+1})$ gives the immediate reward for entering the system state X_{t+1}, which might be negative to indicate a cost rather than a benefit, while the $\sum_{y_{t+1}} p(y_{t+1}|X_{t+1})Q(X_{t+1}, y_{t+1})$ is the expected value of the next state, X_{t+1}, marginalized over the possible next actions y_{t+1}. The expectation of both of these quantities are taken over the possible values that X_{t+1} might take after action y_t is taken in state X_t. $\gamma \in [0, 1]$ is the discount parameter, representing the relative value of future versus immediate rewards. Note that the function is recursive, as the value of a state is defined using the Q function itself and can be calculated using dynamic programming techniques (Howard 1960).

The MDP is often used to solve for an optimal policy $\pi^*(X_t)$, which is the set of actions for a given state X_t that maximize the expected total rewards (Puterman 1994). To model human behavior, however, we view the MDP as a cognitive model (Baker et al. 2009, 2011), for which we assume that people act based on their beliefs, modeled by the state space, action set and transition functions, and in accordance to their desires, which are modeled by the reward structure. In this case the policy is not assumed to be optimal, as humans make mistakes, but instead takes the form a noisy Boltzmann policy (Mahadevan 1996),

$$p(y_t|X_t) \propto \exp\left(\beta Q(X_t, y_t)\right), \tag{12.4}$$

where $\beta \in [0, \infty)$ represents the decision maker's capability to choose actions that will result in higher total rewards. As β increases, the probability of taking the action with the highest Q value, the optimal action, increases. When β goes to zero, the action probabilities become equal, and actions are selected uniformly at random from the action set. Hence, the parameter β controls the degree of randomness in a respondent's actions in way that parallels (the inverse of) temperature in Boltzmann factors. In the present context is can be interpreted as decision-making ability.

12.2.2 Potential for Psychometric Applications

As a cognitive model, the components of the MDP represent the individual's actual goals and beliefs, and it is assumed that the individual's actions are consistent with those goals and beliefs based on the principle of rationality (Baker et al. 2009). The subjective quality of the model allows it to be used for making inferences

Table 12.2 Parameter space for the MDP-MM

Name	Symbol	Type	Interpretation
Goal	R_g	Discrete	Final state attempting to achieve
Motivation	R_m	Continuous	Degree of effort applied to the task
Understanding	T_h	Discrete	Understanding of task dynamics
Capability	β	Continuous	Decision-making capability

about these elements of an agent's cognition, based on observation of their actions (Baker et al. 2011; Rafferty et al. 2015). In particular, inverse reinforcement learning utilizes MDPs to infer discrete goals based on action traces by estimating the most probable values of the reward function (Ng & Russell 2000), while inverse planning algorithms infer student understanding of the effects of their actions by estimating parameters in the transition function (Rafferty et al. 2015).

The MDP measurement framework (MDPMF) (LaMar 2018) provides a structure for making inferences about individuals and populations based on a performance trace through a complex problem space. The parameter space of the MDPMF can be divided into four separable components of the cognitive model (Table 12.2). The *goal* represents what is to be achieved and/or avoided and thus is categorical, while *motivation* is one or more continuous parameters quantifying the amount of effort put into pursuing the goal. *Understanding* represents the model of the task, which includes the state space, the action set, and the transitions functions. While elements of *understanding* might be arguably continuous (e.g. the probability of a particular outcome in the transition function), in this parameterization *understanding* is a set of discrete hypotheses which might represent misconceptions or differing knowledge about the task. Finally *capability* is a continuous parameter that represents ability to optimize decision making given the rest of the model.

Each of these parameters can be estimated at the population or person level, though depending on the quantity of data, it may be prudent to estimate only one of them at the person level. Alternatively, different MDP models can be constructed that represent different cognitive profiles. These might include differing interpretations of the task, different approaches or possible misconceptions. Given two or more contrasting models, the MDPMF can be used to classify performance records based on the model that makes that particular performance more likely.

12.2.3 Code Example

To illustrate the use of the MDPMF we will apply it to a toy problem in which a player has to decide between either fishing or studying every day for 20 days. Each day that you fish you are assumed to catch enough fish to eat for that day. But, in order to study for a day, you must also catch an extra fish to eat. Each day that you study you gain knowledge, and if you gain 5 day's worth of knowledge, you'll be

```
1  library("MDPMF")
2
3  stateSpace = data.frame(Fish=c(0,10), Knowledge=c(0,5), Days=c(0,20))
4  transFun = read.csv("Fishing_Trans.csv")
5  rewards = read.csv("Fishing_Rewards.csv")
6  initState = list(Fish=0,Knowledge=0,Days=20)
7  fishingModel = mdpmf.model(stateSpace, transFun, rewards, initiState)
8  simPlayers = data.frame(PlayerID = 1:500, Beta = rlnorm(500)
9  simPlay = mdpmf.simData(fishingModel, simPlayers)
10
11 # Now use the model to estimate beta from the simulated records
12 betaEst = mdpmf.estBeta(fishingModel, simPlay)
```

Fig. 12.1 Using the `MDPMF` R-package to simulate data and estimate person parameters with the Fishing model

Table 12.3 Transition table for the Fishing MDP

Action	Prob	Fish	Knowledge (K)	Days
Fish	0.5	1	0	−1
Fish	0.5	0	0	−1
Study	1.0	−1	1	−1
SellOne	1.0	−1	0	−1
SellFive	1.0	−5	0	−1

able to get a job with a good salary. You can at any time sell a single fish, or sell five fish, which provides a short term financial reward. The `R` code is shown in Fig. 12.1.

After loading the MDPMF library (line 1), the state space is defined as a data frame with exactly two rows. Each column is a named state-space variable and the first row indicates the minimum value for the variable, while second row indicates the maximum value. All variables are assumed to take on contiguous integer values. For the Fishing problem, the state space is defined by the: number of fish (0–10), amount of knowledge (0–5), and days remaining (0–20) (line 3). The MDP transition function specifies how each available action affect the state space variables using an action-effect table. The transition function is represented within MDPMF as a data table one row for each action outcome (Table 12.3). The first two columns name the action as a string, and the probability of this outcome, as a numeric. The remaining columns indicate, for each state space variable, what effect this action-outcome has on the variable. While not demonstrated here, the probability cells and the state variable effect cells can contain formulas which include the state space variables. In this particular problem, the only non-deterministic result is for the *Fish* action, which has a probability $p_f = 0.5$ of resulting in an extra fish, incrementing the Fish state variable by one. All other actions are deterministic.

Table 12.4 Reward table for the Fishing MDP

Label	Value	Action	Condition
GetJob	25	End	$K \geq 5$
TimeCost	-1	*	1
StudyCost	-1	Study	1
SellOne	2	SellOne	1
SellFive	8	SellFive	1

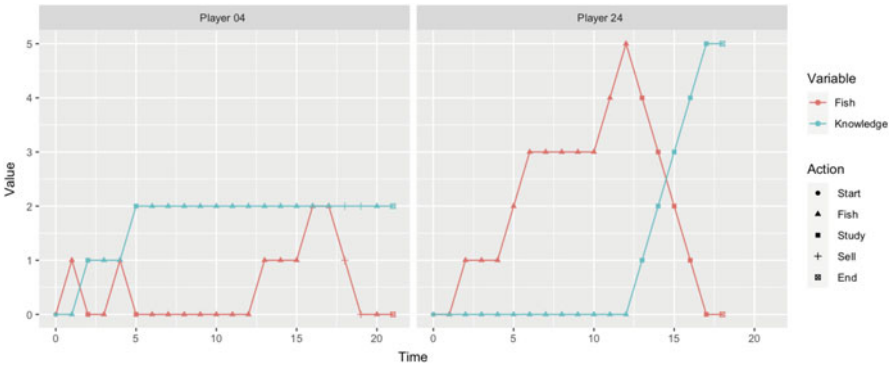


Fig. 12.2 Two simulated play records with Player 04 having low decision-making ability ($\log(\beta) = 0.4$), while Player 24 had a high ability ($\log(\beta) = 3.0$)

The rewards are also specified with a data table (Table 12.4). The four columns indicate reward Label, Value, Action, and StateCondition. For each row the reward value will be achieved whenever the listed action results in a state which complies with the StateCondition. Note that a wildcard "*" can be used to match all actions. For the Fishing problem, we assume a large reward for getting a job, so we give 25 points for ending the game with 5 knowledge units. We also reward selling fish, with rewards of 2 and 8 respectively for selling a single fish and a batch of 5 fish. Each action takes effort so we include a -1 cost for each action taken, but studying is particularly unpleasant so there is an additional -1 cost for studying. Finally, we specify a start state (as a list) of 0 fish, 0 knowledge and 20 days. The model specification files can be downloaded from the website <add book chapter website url>.

To simulate data we also need to specify β_j values for each simulated player. Recall that this parameter describes decision-making ability, and would be a main quantity of interest in psychometric applications. On line 8 a data frame of 500 players is generated with random β_j values, and on line 9 play records are generated based on these simulated players. Records from two simulated players are shown in Fig. 12.2, one with a relatively low β and the other with a very high β. One of the indications of high strategic ability in the second player is the collection of 5 fish before starting to study. This is optimal behavior as it guards against a streak of bad luck in fishing. In the case that the players studies every time they have an extra fish,

as did the player on the left, they might end up running out of time without having enough knowledge to get the high-paying job. The high β player will be able to sell some fish at the end of the 20 days, even if they cannot get the job, while a lower β player would have neither the fish nor the knowledge. One of the advantages of using an MDP model that these strategy differences do not need to be identified in advance, but instead are generated by the model itself.

Once we have simulated play records we can then use these to estimate the β parameters for each player (line 12). If we have each player play the game only one time, we will not have a large quantity of evidence about their strategic capability, but we do have enough to get clear rank ordering among our simulated players.

12.3 State-Space Modeling in Discrete and Continuous Time

State-space models have become an increasingly important tool in behavioral and social sciences. Generically, a state-space model consists of a dynamic model and a measurement model. A measurement model describes how a set of observed variables (e.g., tracked ability estimate of a student) relate to latent *state variables* (e.g., true ability of the student), and a dynamic model portrays how the state variables change over time. When the dynamic and measurement models show regime-dependent behavior, the basic state-space model can be combined with a hidden Markov process further characterized by its initial regime probabilities and transition probabilities between regimes.

12.3.1 Modeling Framework

The dynamic model can either be a discrete-time model as in the one-step-ahead or difference equation form of

$$\boldsymbol{x_i}(t_{i,j+1}) = \boldsymbol{f}_{S_i(t)}\left(\boldsymbol{x_i}(t_{i,j}), t_{i,j}, \boldsymbol{h}_i(t_{i,j})\right) + \boldsymbol{w}_i(t_{i,j+1}), \tag{12.5}$$

or a continuous-time model (i.e., a differential equation model) in the form of

$$d\boldsymbol{x}_i(t) = \boldsymbol{f}_{S_i(t)}\left(\boldsymbol{x}_i(t), t, \boldsymbol{h}_i(t)\right) dt + d\boldsymbol{w}_i(t), \tag{12.6}$$

where i indexes unit of analysis, t indexes time, $\boldsymbol{x}_i(t)$ is a vector of latent variables at time t, $\boldsymbol{h}_i(t)$ is a vector of covariates at time t, and $\boldsymbol{f}_{S_i(t)}(.)$ is a possibly nonlinear dynamic function that depends on a latent categorical variable (i.e., regime indicator), $S_i(t)$. For continuous-time processes, the time intervals between two successive time points may be irregularly spaced, whereas in discrete-time models they are supposed to be equal. In the discrete-time model, $\boldsymbol{w}_i(t)$ is Gaussian distributed process noise with mean zero and regime-specific covariance matrix $\mathbf{Q}_{S_i(t)}$;

in the continuous-time model, $\boldsymbol{w}_i(t)$, is a multivariate Wiener (continuous white noise) process, whose differentials, $d\boldsymbol{w}_i(t)$, are multivariate normally distributed with mean zero and a regime-specific covariance matrix of $\mathbf{Q}_{S_i(t)}$ (Durrett 2010). The initial conditions for the latent continuous variables at the first observed time point $t_{i,1}$ are assumed to follow a normal distribution with mean $\boldsymbol{\mu}_{\boldsymbol{x_1}}$ and covariance matrix, $\boldsymbol{\Sigma}_{\boldsymbol{x_1}}$.

The measurement model defines a linear relation between latent variables $\boldsymbol{x}_i(t_{i,j})$ and observed variables $\boldsymbol{y}_i(t_{i,j})$ as

$$\begin{aligned}\boldsymbol{y}_i(t_{i,j}) = \boldsymbol{\tau}_{S_i(t_{i,j})} + \boldsymbol{\Lambda}_{S_i(t_{i,j})}\boldsymbol{x}_i(t_{i,j}) + \mathbf{A}_{S_i(t_{i,j})}\boldsymbol{h}_i(t_{i,j}) + \boldsymbol{\epsilon}_i(t_{i,j}),\\ \boldsymbol{\epsilon}_i(t_{i,j}) \sim N\left(\mathbf{0}, \boldsymbol{\Sigma}_{\epsilon,S_i(t_{i,j})}\right),\end{aligned} \tag{12.7}$$

where $\boldsymbol{\tau}_{S_i(t_{i,j})}$ is a vector of intercepts, $\boldsymbol{\Lambda}_{S_i(t_{i,j})}$ is a factor loading matrix, $\mathbf{A}_{S_i(t_{i,j})}$ is a matrix of regression coefficients for the time-varying covariates, $\boldsymbol{h}_i(t_{i,j})$, and $\boldsymbol{\epsilon}_i(t_{i,j})$ is a vector of independently and serially uncorrelated normally distributed measurement errors with zero means and a covariance matrix of $\boldsymbol{\Sigma}_{\epsilon,S_i(t_{i,j})}$.

We assume that the change process of $S_i(t_{i,j})$ follows a first-order Markov chain, where the current state only depends on the previous state. The initial regime probabilities for $S_i(t_{i,1})$ and the transition probabilities for $S_i(t_{i,j})$ are represented by two multinomial logistic regression models as

$$\Pr\left(S_i(t_{i,1}) = m|\boldsymbol{h}_i(t_{i,1})\right) \triangleq \pi_{m,i1} = \frac{\exp(a_m + \mathbf{b}_m^T\boldsymbol{h}_i(t_{i,1}))}{\sum_{k=1}^{M}\exp(a_k + \mathbf{b}_k^T\boldsymbol{h}_i(t_{i,1}))}, \tag{12.8}$$

$$\Pr\left(S_i(t_{i,j}) = m|S_i(t_{i,j-1}) = l, \boldsymbol{h}_i(t_{i,j})\right) \triangleq \pi_{lm,it} = \frac{\exp(c_{lm} + \mathbf{d}_{lm}^T\boldsymbol{h}_i(t_{i,j})}{\sum_{k=1}^{M}\exp(c_{lk} + \mathbf{d}_{lk}^T\boldsymbol{h}_i(t_{i,j}))}, \tag{12.9}$$

where M denotes the total number of regimes, $\pi_{m,i1}$ is the probability of individual i's being initially in the regime m, $\pi_{lm,it}$ is the probability of individual i's transitioning from class l at time $t_{i,j-1}$ to class m at time $t_{i,j}$, a_m and c_{lm} denote the logit intercepts, and $\mathbf{b}_m$ and $\mathbf{d}_{lm}$ denote the regression coefficients of the covariates in the associated log-odds (LO) relative to a specified reference regime.[1]

To summarize, the model depicted in Eqs. (12.6)–(12.9) assume that there are group-based laws in how individuals change over time within each regime, and between-person differences stem primarily from (1) changes in $S_i(t_{i,j})$, and (2) potential random effects in the parameters of the dynamic function, $\boldsymbol{f}_{S_i(t)}(.)$, that can be retreated as additional latent variables in the model.

[1] For identification purposes, both Eqs. (12.8) and (12.9) require specification of a reference regime where all parameters in the regression equation are zero.

12.3.2 Potential for Psychometric Applications

In the study of human dynamics, state-space models have been widely used to represent, among other phenomena, attitude transitions (van der Maas et al. 2003), human development (Thatcher 1998; Chow & Nesselroade 2004), physiological regulation (Boker et al. 2014; Brown & Luithardt 1999), and dyadic interactions (Thomas & Martin 1976; Gottman 2002). Regime-switching dynamic models (Kim & Nelson 1999; Hamilton 1989; Muthén & Asparouhov 2011; Chow et al. 2013; Dolan 2009) have been grounded in stagewise development of human cognition and learning processes (Piaget & Inhelder 1969; Dolan et al. 2004; van der Maas & Molenaar 1992; Hosenfeld 1997; Kohlberg & Kramer 1969; van Dijk & van Geert 2007).

For illustration purpose, here we consider a single-regime continuous-time dynamic model of general intelligence development proposed in van der Maas et al. (2006), which we specify as

$$d\boldsymbol{x}_i(t) = d\begin{bmatrix} x_{i1}(t) \\ x_{i2}(t) \end{bmatrix} = \begin{bmatrix} \rho_1 x_{i1}(t)(1 - \frac{x_{i1}(t) - a_{12}x_{i2}(t)}{K}) \\ \rho_2 x_{i2}(t)(1 - \frac{x_{i2}(t) - a_{21}x_{i1}(t)}{K}) \end{bmatrix} dt + \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix} d\boldsymbol{w}_i(t), \tag{12.10}$$

$$\boldsymbol{x}_i(t_{i,1}) \sim N\left(\boldsymbol{\mu}_{\boldsymbol{x_1}} = \begin{bmatrix} \mu_{1,1} \\ \mu_{1,2} \end{bmatrix}, \boldsymbol{\Sigma}_{\boldsymbol{x_1}} = \begin{bmatrix} \sigma^2_{1,1} & \sigma_{1,12} \\ \sigma_{1,12} & \sigma^2_{1,2} \end{bmatrix}\right), \tag{12.11}$$

$$\boldsymbol{y}_i(t_{i,j}) = \boldsymbol{x}_i(t_{i,j}) + \boldsymbol{\epsilon}_i(t_{i,j}), \boldsymbol{\epsilon}_i(t_{i,j}) \sim N\left(\boldsymbol{0}, \boldsymbol{\Sigma}_\epsilon = \begin{bmatrix} \sigma^2_{\epsilon,1} & 0 \\ 0 & \sigma^2_{\epsilon,2} \end{bmatrix}\right), \tag{12.12}$$

such that initial conditions, process noises, and measurement errors are all taken into account and assumed to be normally distributed. Equation (12.10) can be considered as a combination of a logistic growth model and the predator-prey model (Lotka 1925; Volterra 1926; Hofbauer & Sigmund 1988), where the parameters ρ_1 and ρ_2 are growth parameters that determine the steepness of the logistic growth of $x_{i1}(t)$ and $x_{i2}(t)$, parameter K is a carrying capacity parameter that represent limited development potentially due to lack of resources, parameters a_{12} and a_{21} are relation parameters that specify the interaction effects between the latent variables. In the model, we assume $\boldsymbol{y}_i(t)$ consists of two time-varying measures of intelligence, for example, an individual's math and reading scores across time, which are highly correlated. $\boldsymbol{x}_i(t)$ is the underlying skill that are linked to the manifest scores with differences only due to measurement errors. By fitting the model to observed real-time data, we can obtain parameter estimates and infer the underlying developmental process of the hidden skills.

12.3.3 Numeric Example with R Code

There are a lot of packages for estimating multivariate discrete-time or continuous-time state-space models, including R packages such as dlm (Petris et al. 2009; Petris 2010), dse (Gilbert 2006 or later), FKF (Luethi et al. 2014), KFAS (Helske 2017), MARSS (Holmes et al. 2012, 2013), bsts (Scott 2017), ctsem (Driver et al. 2017), and the OpenMx 2.0 release (Neale et al. 2016), and other implementations such as the MKFM6 (Dolan 2005), the SsfPack (Koopman et al. 1999), and the Mplus (Muthén & Muthén 1998–2012). However, they are mostly only used for linear dynamic models, and often lack utilities for nonlinear and regime-switching dynamic models. On the other hand, the dynr R package can be utilized for fitting more general nonlinear discrete-time and continuous-time dynamical models potentially with regime switching (Ou et al. 2017). The dynr package allows users to specify the models in the familiar R language, while perform quick and efficient computations in C. Here, we briefly illustrate the use of dynr to fit the proposed model to a set of simulated data.[2] We summarize the input code and output in Fig. 12.3.

The first step is to structure the data with the dynr.data() function. Next, we specify the various parts (i.e., "recipes") of the model using prep.*() functions. The functions prep.measurement(), prep.formulaDynamics(), and prep.noise() respectively specifies the measurement model, the dynamic functions, and the noise covariance matrices in both Eq. (12.10) and (12.12). These covariance matrices need to be positive definite. The values.* arguments provide the starting values and fixed values of the components or parameters of the model, whereas the params.* arguments give the free parameter names corresponding to the values or indicate a fixed value by using the reserved name "fixed." For example, the values.load argument of prep.measurement() is the fixed identity factor loading matrix in Eq. (12.12). The first argument, formula, of prep.formulaDynamics() is a list of formulas. Each element in the list is a single, univariate, formula that defines a differential (if isContinuousTime = TRUE) or difference (if isContinuousTime = FALSE) equation. There should be one formula for every latent variable, indicated by the left-hand side of each formula and in the order in which the latent variables are specified by the state.names argument in prep.measurement(). The right-hand side of each formula gives a (possibly nonlinear) function that may involve free parameters, numerical constants, exogenous covariates, and other arithmetic/mathematical functions that define the dynamics of the latent variables. The startval argument is a named vector giving the names of the free parameters and their starting values. The isContinuousTime argument is a binary flag that defines the switch between

[2]The data were simulated using the R package Sim.DiffProc (Guidoum & Boukhetala 2016) with true parameters: $\rho_1 = 0.2$, $\rho_2 = 0.4$, $a_{12} = -0.1$, $a_{21} = -0.3$, $K = 5$, $\sigma_1^2 = \sigma_2^2 = \sigma_{\epsilon,1}^2 = \sigma_{\epsilon,2}^2 = 0.01$, $\mu_{1,1} = \mu_{1,2} = 2$, $\sigma_{1,1}^2 = \sigma_{1,2}^2 = 0.1$, $\sigma_{1,12} = 0$.

Input

```
#---- Load the package ----
require(dynr)
#---- Read in the data ----
data <- dynr.data(data.simulate, id = "id", time = "time", observed = c("x", "y"))

#---- Starting Values ----
startVals <- c(rho1 = .2, rho2 = .2, a12 = 0, a21 = 0, K = 3,
               var1 = 0.01, var2 = 0.01, var_e1 = .01, var_e2 = 0.01,
               muOne1 = 2, muTwo1 = 2, varOne1 = .05, varTwo1 = .05, cov_OneTwo1 = 0)

#---- Prepare the recipes ----
# Dynamic Model
formula =list(One ~ rho1 * One * (1 - (One + a12 * Two)/K),
              Two ~ rho2 * Two * (1 - (Two + a21 * One)/K))
dynm <- prep.formulaDynamics(formula = formula,
                             startval = c(rho1 = unname(startVals["rho1"]), rho2 = unname(startVals["rho2"]),
                                          a12 = unname(startVals["a12"]), a21 = unname(startVals["a21"]),
                                          K = unname(startVals["K"])),
                             isContinuousTime = TRUE)
# Measurement Model
meas <- prep.measurement(
  values.load = matrix(c(1, 0,
                         0, 1), ncol = 2, byrow = TRUE),
  obs.names = c("x", "y"), state.names = c("One", "Two"))

# Initial Conditions
initial <- prep.initial(
  values.inistate = c(startVals["muOne1"], startVals["muTwo1"]),
  params.inistate = c("muOne1", "muTwo1"),
  values.inicov = matrix(c(startVals["varOne1"], startVals["cov_OneTwo1"],
                           startVals["cov_OneTwo1"], startVals["varTwo1"]), 2, 2, byrow = TRUE),
  params.inicov = matrix(c("varOne1", "cov_OneTwo1",
                           "cov_OneTwo1", "varTwo1"), 2, 2, byrow = TRUE))
# Noise Covariance Matrix
mdcov <- prep.noise(
  values.latent=diag(c(startVals["var1"], startVals["var2"])),
  params.latent=diag(c("var1", "var2")),
  values.observed=diag(c(startVals["var_e1"], startVals["var_e2"])),
  params.observed=diag(c("var_e1","var_e2")))
#---- Cooking materials ----
# Put all the recipes together in a Model Specification
model <- dynr.model(dynamics = dynm, measurement = meas,
                    noise = mdcov, initial = initial,
                    data = data, outfile="mutualism.c")
# Estimate free parameters
res <- dynr.cook(dynrModel = model)
#---- Examine results ----
summary(res, digits = 2)

dynr.ggplot(res, model, style = 2, title = "Results of fitting the model to simulated data",
            numSubjDemo = 3, text = element_text(size = 16))
```

Output

```
> summary(res, digits = 2)
Coefficients:
             Estimate Std. Error t value   ci.lower   ci.upper Pr(>|t|)
rho1        0.1986368  0.0052838  37.593  0.1882807  0.2089929   <2e-16 ***
rho2        0.4007443  0.0042071  95.255  0.3924986  0.4089900   <2e-16 ***
a12        -0.1048686  0.0151351  -6.929 -0.1345329 -0.0752044   <2e-16 ***
a21        -0.3015137  0.0205265 -14.689 -0.3417449 -0.2612825   <2e-16 ***
K           4.9966545  0.0909453  54.941  4.8184049  5.1749041   <2e-16 ***
var1        0.0106179  0.0007077  15.003  0.0092308  0.0120050   <2e-16 ***
var2        0.0109027  0.0007680  14.196  0.0093974  0.0124081   <2e-16 ***
var_e1      0.0098775  0.0002675  36.925  0.0093532  0.0104018   <2e-16 ***
var_e2      0.0098144  0.0002723  36.037  0.0092806  0.0103481   <2e-16 ***
muOne1      1.9512467  0.0317116  61.531  1.8890931  2.0134003   <2e-16 ***
muTwo1      2.0639135  0.0346952  59.487  1.9959122  2.1319148   <2e-16 ***
varOne1     0.0966144  0.0141652   6.821  0.0688511  0.1243778   <2e-16 ***
cov_OneTwo1 0.0091024  0.0109908   0.828 -0.0124392  0.0306440    0.204
varTwo1     0.1164115  0.0169692   6.860  0.0831524  0.1496706   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

-2 log-likelihood value at convergence = -12923.67
AIC = -12895.67
BIC = -12804.43
```

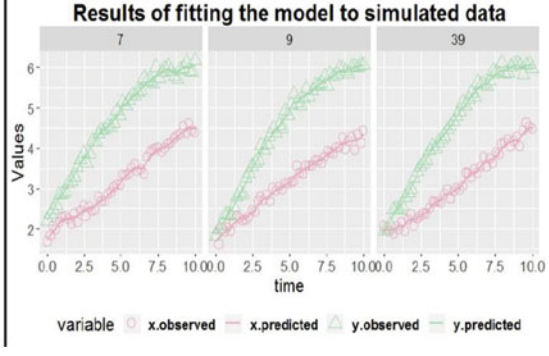


Fig. 12.3 Numeric example with `dynr`

continuous- and discrete-time modeling. The initial conditions as in Eq. (12.11) are specified through the `prep.initial()` function.

After the recipes for all parts of the model are defined, the `dynr.model()` function creates the model and stores it in a `dynrModel` object. The function requires `dynamics`, `measurement`, `noise`, `initial`, and `data` as mandatory inputs for all models. When there are multiple regimes in the model, the `regimes` argument should be provided. When parameters are subject to transformation functions, a `transform` argument can be added. The `dynr.model()` function takes the recipes and the data, and combines information from both to write a `C` function. The model is then "cooked" with the `dynr.cook()` function to estimate the free parameters and their standard errors. When cooking, the `C` code that was written by `dynr.model()` is compiled on the user's end and dynamically linked to the rest of the compiled `dynr` code. Then the backend `C` program is executed to compute a negative log-likelihood function and optimize it for parameter estimation. After the model has been cooked, the `summary()` method provides a table of free parameter names, estimates, standard errors, t-values, and Wald-type confidence intervals. The `dynr.ggplot()` method creates a plot of the predicted and observed state estimates. In Fig. 12.3, the line plots show the predicted (line) versus observed (point) values for two variables, X and Y, and each panel depicts this information for a randomly selected process. Comparing the predicted to observed values is one way to assess the fit of the model to the data, which in this case is acceptable.

12.4 Temporal Point Processes

Temporal point processes are used to model statistical dependence among events that occur at irregular time intervals (Cox & Isham 1980; Daley & Vera-Jones 2003). While point processes have been used widely, their application to human dynamics in computer-mediated settings is of particular relevance (Barabási 2005; Blundell et al. 2012; Chen & Tan 2018; Crane & Sornette 2008; Fox 2015; Halpin & De Boeck 2013; Karsai et al. 2011; Matsubara et al. 2012; Oliveira & Vazquez 2009). In this research, the events of interest are typified by sending emails, texting, tweeting, chatting, visiting web pages, watching videos, clicking buttons on web pages, and so on.

Similar to survival analyses, temporal point processes can be thought of as a general framework for modeling time-to-event data, and the central concept of a hazard function is shared by both approaches. Unlike survival analyses, point processes allow the probability of an event to depend on past events, so the hazard function is defined conditional on the history of the process.

12.4.1 Model Specification

There are several equivalent ways of characterizing point processes (Brillinger et al. 2002). Here we focus on their representation as a finite, integer-valued random measure, $N([a, b))$, which counts the number of events falling in the half-open time interval $[a, b)$. The notation $dN(t) \equiv N([t, t + dt))$ is used to denote the number of events occurring in an infinitesimal interval and the requirement that $\Pr\{dN(t) > 1\} = o(dt)$ defines an *orderly process* as one in which the probability of more than one event occurring at once is negligible. We also let t_i, $i \in \mathbb{N}$, denote the sequence of times for which $dN(t) = 1$.

Models for orderly temporal point processes can be specified in terms of the conditional intensity function (CIF) (Daley & Vera-Jones 2003, Chap. 7), which describes the "instantaneous" Bernoulli probability of an event at time t (i.e., the hazard rate) as conditional on the history of the process, H_t:

$$\lambda(t \mid H_t) \equiv \lim_{dt \to 0} \frac{\Pr\{dN(t) = 1 \mid H_t\}}{dt}. \tag{12.13}$$

H_t is formally defined as a filtration on the underlying probability space of the process, but is more intuitively understood as the set previous events, $H_t = \{t_i \mid t_i < t\}$. In the multivariate case, $N([a, b))$ is vector-valued and each univariate margin describes the occurrence of a different type of event in the time period $[a, b)$. Additional information about events can also be encoded as covariates or "marks". In this case, each event is described by a vector of covariates X_i in addition to t_i, and the resulting joint CIF is factorized into the CIF for the event times and the conditional distribution of the marks (Daley & Vera-Jones 2003, Chap. 7):

$$\lambda(t, X \mid H_t^*) \equiv \lambda(t \mid H_t^*) \times \Pr\{X \mid t, H_t^*\}. \tag{12.14}$$

The history of the process $H_t^* = \{(t_i, X_i) \mid t_i < t\}$ now includes the event times preceding time t as well as their marks.

A wide variety of statistical models for temporal point processes have been proposed. The stationary Poisson process is defined such that (a) the number of points falling in any time interval has a Poisson distribution and (b) non-overlapping intervals are statistically independent. From a substantive perspective, the stationary Poisson process is used to represent "white noise," or independently and identically distributed event times. A general distinction is made between processes that are over-dispersed or under-dispersed, relative to the Poisson. Over-dispersed or *excitatory* processes exhibit positive serial dependence (e.g., a positive serial correlation); under-dispersed or *inhibitory* process exhibit negative serial dependence.

Empirical research has supported the conclusion that human dynamics in computer-mediated settings are typically excitatory (Barabási 2005). The Hawkes

process (Hawkes 1971) is a general framework for modeling excitatory processes, playing a role similar to autoregressive models in conventional time series analysis (Daley & Vera-Jones 2003, p. 183). The CIF of the (univariate, unmarked) Hawkes process can be written as (Hawkes 1971, eq. 8)

$$\lambda(t \mid H_t) = \mu + \sum_{t_i < t} \phi(t - t_i) \tag{12.15}$$

where $\mu > 0$ is a baseline, which can be a function of time but is here treated as a constant, and $\phi : \mathbb{R}_+ \to \mathbb{R}_+$ is the response function that governs how the process depends on its past. The requirement that $\phi(u) \geq 0$ implies that the occurrence of an event has a non-negative influence on the probability future events, giving the process its excitatory character. The further requirement $\int_0^\infty \phi(u)d(u) < 1$ ensures that the process is stationary (Hawkes 1971). Together these requirements imply that we may write $\phi(u) = \alpha\, f(u)$ where f is a density on $\mathbb{R}+$, which we refer to as the response kernel, and $\alpha \in [0, 1)$ is referred to as the response intensity (also known as the "branching ratio"). This factorization suggests the following two-part strategy for parametric modeling.

12.4.1.1 The Response Kernel

The response kernel characterizes the *short-term dynamics* immediately following an event t_i, such as the latency period and the rate of decay. The mathematics of the process are especially tractable when the response kernel is chosen to be the exponential distribution (Hawkes 1971; Liniger 2009), and this specification is commonly used in statistical applications. However, our own view is that the choice of response kernel is a relatively strong assumption about human dynamics, and it should therefore be evaluated with respect to empirical data rather than assumed a priori. This point is illustrated in the numerical example (see Sect. 12.4.3).

12.4.1.2 The Response Intensity

The relation $\alpha = \int_0^\infty$ follows directly from the two-part factorization of the response function and motivates the interpretation of the response intensity as describing the *overall* or *long-term dynamics* of the Hawkes process. Importantly, this interpretation is independent of the particular form of the response kernel—i.e., independent of the short-term dynamics of the process. In many applications, it is useful to let the response intensity depend on covariates, which in turn can lead to various modeling choices via the marking distribution (see Eq. (12.14)). As we now discuss, this makes the response intensity a natural candidate for describing person-level attributes in psychometric settings.

12.4.2 Potential for Psychometric Applications

Process data obtained from computer-based assessments often take the form of time-stamped user activity logs. These data are (a) events that occur at irregular time intervals, (b) with different types of events characterized by different features and (c) sequential independence among events being highly implausible. Point processes provide a relatively natural framework for such data. They are specifically designed to model temporal dependence among events that occur at irregular time intervals and readily incorporate time-varying covariates. Moreover, the response intensity parameter provides a convenient way of characterizing respondents' overall or long-term temporal dynamics. For example, Halpin et al. (2017) addressed the utility of the bivariate Hawkes process for modeling computer-based educational assessments involving collaboration among students (also see Halpin 2016; von Davier & Halpin 2013). They derived a number of individual- and group-level indices based on the response intensity; when applied to students' online chat logs, these indices were far more predictive of task outcomes than more commonly used quantitative summaries based static features of chat dialogue (e.g., word counts).

While temporal point processes hold promise for modeling process data, there remains substantial work to be done before they are optimized for assessment applications. First, reliable estimation of the focal parameters requires relatively long time series. Because test duration is necessarily limited, we must improve the efficiency of current estimation methods. Second, analytical results on standard errors exist only for relatively simple models (Ozaki 1979), or provide asymptotic bounds rather than exact equalities (Halpin et al. 2017). Such results play a role analogous to the item- and test-information functions used in IRT—they allow for the design of tests that satisfy a desired level of statistical precision, thereby providing a basis for reliably assessing individual differences. Third, current approaches are useful for modeling dependence among the actions of one or more respondents, but are less well suited to describing how that dependence is conditional on (i.e., moderated by) task features.

12.4.3 Numeric Example with `R` Code

Analysis of temporal point process using the `R` language is supported by a number of packages, notably `ptproc` (Peng 2003) and `ptprocess` (Harte 2010). The simple example presented here uses custom functions in combination with R's generic optimizer `optim`. The custom functions allow for simulation and estimation of a univariate Hawkes process with gamma response kernel, and are available with the online materials for this chapter (Fig. 12.4).

The example code shown in the left hand side of Fig. 12.4 simulates 10,000 events from a univariate, unmarked Hawkes process with a two-parameter gamma response kernel (model parameters given in Line 4). The script then estimates the

Input

```
source(hawkes_functions.R)

# Data generation: 10,000 events from Hawkes process with gamma kernel
parms <- list("mu" = .1, "alpha" = .6, "shape" = 15, "scale" = .5)
set.seed(1001)
times <- hawkes_sim(10000, parms)

# Fit Hawkes model with gamma and exponential kernels
gamma_hawkes <- optim(unlist(parms),
   logl, times = times, method = "L-BFGS-B",
   lower = rep(.00001, 4), upper = c(1, 1, 20, 20),
   hessian = T)

exp_hawkes <- optim(unlist(parms[-3]),
   logl, times = times, method = "L-BFGS-B",
   lower = rep(.00001, 3), upper = c(1, 1, 20),
   hessian = T)

# Likelihood ratio test of model fit
lr <- 2 * (exp_hawkes$value - gamma_hawkes$value)
cat("Likelihood ratio test of nested models. ",
   "LR: ", lr, "p-value: ", pchisq(lr, 1, lower.tail = F))

# 95% confidence intervals on overall intensity parameter
gamma_alpha <- gamma_hawkes$par["alpha"]
gamma_alpha_se <- se(gamma_hawkes)["alpha"]
cat("95% CI on overall intensity, gamma kernel. ",
   "Lower: ",  gamma_alpha - gamma_alpha_se * 1.96,
   "Upper: ",  gamma_alpha + gamma_alpha_se * 1.96)

exp_alpha <- exp_hawkes$par["alpha"]
exp_alpha_se <-  se(exp_hawkes)["alpha"]
cat("95% CI on overall intensity, exponential kernel. ",
   "Lower: ",  exp_alpha - exp_alpha_se * 1.96,
   "Upper: ",  exp_alpha + exp_alpha_se * 1.96)
```

Output (abbreviated)

```
Lines 21-22:
Likelihood ratio test of nested models.
LR: 1034.912
p-value: 4.63237e-227

Lines 27-29:
95% CI on overall intensity, gamma
kernel.
Lower: 0.5774396
Upper: 0.6263645

Lines 33-35:
95% CI on overall intensity,
exponential kernel.
Lower: 0.6108748
Upper: 0.6654887
```

Fig. 12.4 Numeric example

Hawkes process for both the gamma and the exponential response kernels. The right hand side of the Figure summarize the relevant output. The likelihood ratio test shows that the (correctly specified) gamma response kernel fit the data better than the (misspecified) exponential. More importantly, the estimate of the response intensity was notably biased using the misspecified exponential response kernel. Using the gamma kernel, all parameters were recovered to 4 decimal places.

This simple example demonstrates that parameter recovery of the long-term dynamics of the Hawkes process (i.e., the response intensity parameter) is affected when the short-term dynamics (i.e., the response kernel) are misspecified. In terms of practical implications, this suggests that careful consideration should be given to specification of the response kernel when applying the Hawkes process, rather than simply assuming the exponential kernel for its mathematical convenience.

12.5 Conclusion

This chapter has summarized three modeling frameworks that exemplify three different ways of conceptualizing serial dependence in the analysis process data: as purely sequential, in discrete time, or in continuous time. Of course, there is much more to be said about the analysis of time series data, but we hope that this short chapter has provided the reader with the general flavor of these three

main approaches and an indication of the breadth of models that are available. Currently, it seems that the sequential approach has seen relatively wide application in psychometric settings, for example in tree-like decision models (Cho et al. 2020; Jeon & De Boeck 2016) or machine learning approaches that address sequence-to-sequence problems (Qiao & Jiao 2018; Tang et al. 2020). When time is considered explicitly, it is most often as response times for individual items, which do not invoke a notion of serial dependence (De Boeck & Jeon 2019). Nonetheless, we are optimistic that continued research will lead to many applications for discrete and continuous time series in psychometric settings.

We close this chapter by suggesting three main future research directions that seem pertinent to the analysis of psychometric process data, regardless of the specific modeling approach used. The first is topic assessment design. Too often, process data are merely a by-product of using technology to deliver assessments. While process data surely can reveal important aspects of respondents' in-task behaviors, it would be surprising if this happened by accident. A first step towards realizing the transformative potential of process data is to design assessments so that the data have a clear meaning.

The second topic is sample efficiency. Time series models such as those considered in this chapter are typically designed for "small N, large T" settings—i.e., a small number of units observed over a long duration of time. By contrast, assessment settings are decidedly large N and, given practical constraints, moderate T at best. Consequently we suggest that future methodological research should address adaptations of time series methods to applications involving many parallel data streams of relatively short duration. In particular, random effects extension of existing time series models could serve this purpose, and also provide much of the machinery available from conventional psychometric methodology (e.g., population-specific parameter calibration, improved efficiency of parameter estimates, and empirical Bayes' approaches to scoring).

Third, there is much empirical research to be done on the reliability, validity, and fairness of inferences made from process data. While the psychometric community has so far been willing to make use of process data for studying the properties of existing assessment materials, it would seem that the real excitement about these data lie in their potential for enhancing the kinds of inferences we can make about respondents' knowledge, skills, and other attributes. While this may yet be a long way off, we think that statistical models that incorporate serial dependence will play an important role in reaching this goal.

References

Baker, C., Saxe, R., & Tenenbaum, J. (2009). Action understanding as inverse planning. *Cognition, 113*(3), 329–349.

Baker, C., Saxe, R., & Tenenbaum, J. (2011). Bayesian theory of mind: Modeling joint beliefdesire attribution. In *Proceedings of the Thirty-Third Annual Conference of the Cognitive Science Society* (pp. 2469–2474).

Barabási, A.-L. (2005, May). The origin of bursts and heavy tails in human dynamics. *Nature, 435*, 207–211. https://doi.org/10.1038/nature03526.1

Blundell, C., Heller, K. A., & Beck, J. M. (2012). Modelling reciprocating relationships with Hawkes processes. *Advances in Neural Information Processing Systems, 25*, 2600–2608.

Boker, S. M., Neale, M. C., & Klump, K. L. (2014). A differential equations model for the ovarian hormone cycle. In P. C. M. Molenaar, R. M. Lerner, & K. M. Newell (Eds.), *Handbook of developmental systems theory and methodology* (pp. 369–391). New York, NY: Guilford Press.

Brillinger, D. R., Guttorp, P. M., & Schoenberg, F. P. (2002). Point processes, temporal. In A. H. El-Shaarawi & W. W. Piegorsch (Eds.), *Encyclopedia of environmetrics* (Vol. 3, pp. 1577–1581). Chichester, England: Wiley.

Brown, E. N. & Luithardt, H. (1999). Statistical model building and model criticism for human circadian data. *Journal of Biological Rhythms, 14*, 609–616. https://doi.org/10.1177/074873099129000975

Chen, F., & Tan, W. H. (2018). Marked self-exciting point process modelling of information diffusion on twitter. Preprint, pp. 1–18. arXiv: 1802.09304

Cho, S.-J., Brown-Schmidt, S., Boeck, P. D., & Shen, J. (2020). Modeling intensive polytomous time-series eye-tracking data: A dynamic tree-based item response model. *Psychometrika, 85*(1), 154–184. https://doi.org/10.1007/s11336-020-09694-6

Chow, S.-M. & Nesselroade, J. R. (2004). General slowing or decreased inhibition? Mathematical models of age differences in cognitive functioning. *Journals of Gerontology B, 59*(3), 101–109. https://doi.org/10.1093/geronb/59.3.P101

Chow, S.-M., Grimm, K. J., Guillaume, F., Dolan, C. V., & McArdle, J. J. (2013). Regimeswitching bivariate dual change score model. *Multivariate Behavioral Research, 48*(4), 463–502. https://doi.org/10.1080/00273171.2013.787870

Cox, D. R., & Isham, V. (1980). *Point processes*. New York: Chapman Hall/CRC.

Crane, R., & Sornette, D. (2008). Robust dynamic classes revealed by measuring the response function of a social system. *Proceedings of the National Academy of Sciences, 105*(41), 15649–15653. https://doi.org/10.1073/pnas.0803685105

Gilbert, P. (2006 or later). *Brief user's guide: Dynamic systems estimation*. Retrieved from http://cran.r-project.org/web/packages/dse/vignettes/Guide.pdf

Daley, D. J., & Vera-Jones, D. (2003). *An introduction to the theory of point processes: Elementary theory and methods (Second)*. New York: Springer.

De Boeck, P., & Jeon, M. (2019). An overview of models for response times and processes in cognitive tests. *Frontiers in Psychology, 10*. https://doi.org/10.3389/fpsyg.2019.00102

Dolan, C. V. (2005). MKFM6: Multi-group, multi-subject stationary time series modeling based on the Kalman filter. Retrieved from http://users.fmg.uva.nl/cdolan/

Dolan, C. V. (2009). Structural equation mixture modeling. In R. E. Millsap & A. Maydeu-Olivares (Eds.), *The SAGE handbook of quantitative methods in psychology* (pp. 568–592). Thousand Oaks, CA: Sage.

Dolan, C. V., Jansen, B. R., & Van der Maas, H. L. J. (2004). Constrained and unconstrained multivariate normal finite mixture modeling of Piagetian data. *Multivariate Behavioral Research, 39*(1), 69–98. https://doi.org/10.1207/s15327906mbr3901_3

Driver, C. C., Oud, J. H. L., & Voelkle, M. C. (2017). Continuous time structural equation modelling with R package **ctsem**. *Journal of Statistical Software, 77*(5), 1–35. https://doi.org/10.18637/jss.v077.i05

Durrett, R. (2010). *Probability: Theory and examples* (4th ed.). Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge, New York: Cambridge University Press.

Ercikan, K., & Pellegrino, J. W. (2017). *Validation of score meaning for the next generation of assessments: The use of response processes*. New York: Routledge.

Fox, E. W. (2015). *Estimation and Inference for Self-Exciting Point Processes with Applications to Social Networks and Earthquake Seismology*. Dissertation Thesis, University of Los Angeles.

Gottman, J. M. (2002). *The mathematics of marriage: Dynamic nonlinear models*. Cambridge, MA: The MIT Press.

Guidoum, A. C., & Boukhetala, K. (2016). SIM.DIFFPROC: Simulation of diffusion processes. R package version 3.6. Retrieved from https://cran.r-project.org/package=Sim.DiffProc

Halpin, P. (2016). *Psychometric Analysis of the Child Functioning Module: Comparisons Between Serbia and Mexico*. UNICEF, Division of Data, Research, and Policy.

Halpin, P. F., & De Boeck, P. (2013). Modelling dyadic interaction with Hawkes processes. *Psychometrika, 78*(4), 793–814. https://doi.org/10.1007/s11336-013-9329-1

Halpin, P. F., von Davier, A. A., Hao, J., & Liu, L. (2017). Measuring student engagement during collaboration. *Journal of Educational Measurement, 54*(1), 70–84. https://doi.org/10.1111/jedm.12133

Hamilton, J. D. (1989). A new approach to the economic analysis of nonstationary time series and the business cycle. *Econometrica, 57*, 357–384. https://doi.org/10.2307/1912559

Harte, D. (2010). PtProcess: An R package for modelling marked point processes indexed by time. *Journal of Statistical Software, 35*(8). https://doi.org/10.18637/jss.v035.i08

Hawkes, A. G. (1971). Point spectra of some mutually exciting point processes. *Journal of the Royal Statistical Society, Series B, 33*(3), 438–443. https://doi.org/10.1073/pnas.0703993104. arXiv:1011.1669v3

Helske, J. (2017). KFAS: Exponential family state space models in R. *Journal of Statistical Software, 78*(10), 1–39. https://doi.org/10.18637/jss.v078.i10

Hofbauer, J., & Sigmund, K. (1988). *The theory of evolution and dynamical systems: Mathematical aspects of selection (London Mathematical Society Student Texts)*. Cambridge: Cambridge University Press. Retrieved from http://www.worldcat.org/isbn/0521358388

Holmes, E., Ward, E., & Wills, K. (2013). *MARSS: Multivariate autoregressive state-space modeling*. R package version 3.9. Retrieved from http://cran.r-project.org/web/packages/MARSS/

Holmes, E. E., Ward, E. J., & Wills, K. (2012). MARSS: Multivariate autoregressive statespace models for analyzing time-series data. *The R Journal, 4*(1), 30.

Hosenfeld, B. (1997). Indicators of discontinuous change in the development of analogical reasoning. *Journal of Experimental Child Psychology, 64*, 367–395. https://doi.org/10.1006/jecp.1996.2351

Howard, R. A. (1960). *Dynamic programming and Markov processes* (1st ed.). Cambridge, MA: The MIT Press.

Jeon, M., & De Boeck, P. (2016). A generalized item response tree model for psychological assessments. *Behavior Research Methods, 48*(3), 1070–1085. https://doi.org/10.3758/s13428-015-0631-y

Karsai, M., Kivelä, M., Pan, R. K., Kaski, K., Kertész, J., Barabási, a.-L., & Saramäki, J. (2011). Small but slow world: How network topology and burstiness slow down spreading. *Physical Review E, 83*(2), 025102. https://doi.org/10.1103/PhysRevE.83.025102

Kim, C.-J., & Nelson, C. R. (1999). *State-space models with regime switching: Classical and Gibbs-sampling approaches with applications*. Cambridge, MA: MIT Press.

Kohlberg, L., & Kramer, R. (1969). Continuities and discontinuities in childhood and adult moral development. *Human development, 12*(2), 93–120. https://doi.org/10.1159/000270857

Koopman, S. J., Shephard, N., & Doornik, J. A. (1999). Statistical algorithms for models in state space using **ssfpack** 2.2. *Econometrics Journal*, *2*(1), 113–166. https://doi.org/10.1111/1368-423X.00023

LaMar, M. M. (2018). Markov decision process measurement model. *Psychometrika, 83*, 67–88. https://doi.org/10.1007/s11336-017-9570-0

Liniger, T. (2009). *Multivariate Hawkes Processes*. Dissertation Thesis, Swiss Federal Institute of Technology. https://doi.org/10.3929/ethz-a-006037599

Lord, F., & Novick, M. (1968). *Statistical theories of mental test scores*. Addison-Wesley series in behavioral science. Addison-Wesley Pub. Co. Retrieved from https://books.google.com/books?id=FW5EAAAAIAAJ

Lotka, A. J. (1925). *Elements of physical biology*. Baltimore, MD: Williams & Wilkins.

Luethi, D., Erb, P., & Otziger, S. (2014). *FKF: Fast Kalman filter*. R package version 0.1.3. Retrieved from https://CRAN.R-project.org/package=FKF

Mahadevan, S. (1996). Average reward reinforcement learning: Foundations, algorithms, and empirical results. *Machine Learning, 22*, 159–195.

Matsubara, Y., Sakurai, Y., Prakash, B. A., Li, L., & Faloutsos, C. (2012). Rise and fall patterns of information diffusion: Model and implications. In *KDD '12: Proceedings of the 18th ACM SIGKDD* (pp. 6–14). New York, NY: ACM Press.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature, 518*(7540), 529–533.

Muthén, B. O. & Asparouhov, T. (2011). *LTA in **Mplus**: Transition probabilities influenced by covariates*. **Mplus** Web Notes: No. 13. Retrieved from http://www.statmodel.com/examples/LTAwebnote.pdf

Muthén, L. K., & Muthén, B. O. (1998–2012). *Mplus user's guide*(7th ed.). Los Angeles, CA: Muthén & Muthén.

Neale, M. C., Hunter, M. D., Pritikin, J. N., Zahery, M., Brick, T. R., Kirkpatrick, R. M., ... Boker, S. M. (2016). OPENMX2.0: Extended structural equation and statistical modeling. *Psychometrika, 80*(2), 535–549. https://doi.org/10.1007/s11336-014-9435-8

Ng, A. Y., & Russell, S. (2000). Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning (2000)* (pp. 663–670).

Oliveira, J. G., & Vazquez, A. (2009). Impact of interactions on human dynamics. *Physica A: Statistical Mechanics and Its Applications, 388*(2–3), 187–192. https://doi.org/10.1016/j.physa.2008.08.022. arXiv: physics/0603064

Ou, L., Hunter, M. D., & Chow, S.-M. (2017). DYNR: Dynamic modeling in R. R package version 0.1.11-5.

Ozaki, T. (1979). Maximum likelihood estimation of Hawkes' self-exciting point processes. *Annals of the Institute of Statistical Mathematics, 31*(1), 145–155. https://doi.org/10.1007/BF02480272

Peng, R. D. (2003). Multi-Dimensional Point Process Models in R. *Journal of Statistical Software, 8*(16). https://doi.org/10.18637/jss.v008.i16

Petris, G. (2010). An R package for dynamic linear models. *Journal of Statistical Software, 36*(12), 1–16. Retrieved from http://www.jstatsoft.org/v36/i12/

Petris, G., Petrone, S., & Campagnoli, P. (2009). *Dynamic linear models with R. useR!*. New York, NY: Springer.

Piaget, J., & Inhelder, B. (1969). *The psychology of the child*. New York, NY: Basic Books.

Puterman, M. L. (1994). *Markov decision processes: Discrete stochastic dynamic programming*. New York: Wiley.

Qiao, X., & Jiao, H. (2018). Data mining techniques in analyzing process data: A didactic. *Frontiers in Psychology, 9*. https://doi.org/10.3389/fpsyg.2018.02231

Rafferty, A. N., LaMar, M. M., & Griffiths, T. L. (2015). Inferring learners' knowledge from their actions. *Cognitive Science, 39*(3), 584–618.

Russell, S., & Norvig, P. (2009). *Artificial intelligence: A modern approach* (3rd ed.). Upper Saddle River: Pearson.

Rust, J. (1994). Structural estimation of Markov decision processes. In R. Engle & D. Mc-Fadden (Eds.), *Handbook of econometrics* (Vol. 4, pp. 3081–3143). Elsevier Science.

Scott, S. L. (2017). *BSTS: Bayesian structural time series*. R package version 0.7.1. Retrieved from https://CRAN.R-project.org/package=bsts

Tang, X., Zhang, S., Wang, Z., Liu, J., & Ying, Z. (2020). ProcData: An R Package for Process Data Analysis. arXiv:2006.05061 [cs, stat].

Thatcher, R. W. (1998). A predator-prey model of human cerebral development. In K. M. Newell & P. C. M. Molenaar (Eds.), *Applications of nonlinear dynamics to developmental process modeling* (pp. 87–128). Mahwah, NJ: Lawrence Erlbaum.

Thomas, E. A., & Martin, J. A. (1976). Analyses of parent-infant interaction. *Psychological Review, 83*(2), 141–156. https://doi.org/10.1037/0033-295X.83.2.141

van der Maas, H. L. J. & Molenaar, P. C. M. (1992). Stagewise cognitive development: An application of catastrophe theory. *Psychological Review, 99*(3), 395–417. https://doi.org/10.1037//0033-295x.99.3.395

van der Maas, H. L. J., Kolstein, R., & van der Pligt, J. (2003). Sudden transitions in attitudes. *Sociological Methods & Research, 32*, 125–152. https://doi.org/10.1177/0049124103253773

van der Maas, H. L., Dolan, C. V., Grasman, R. P., Wicherts, J. M., Huizenga, H. M., & Raijmakers, M. E. (2006). A dynamical model of general intelligence: The positive manifold of intelligence by mutualism. *Psychological Review, 113*(4), 842.

van Dijk, M., & van Geert, P. (2007). Wobbles, humps and sudden jumps: A case study of continuity, discontinuity and variability in early language development. *Infant and Child Development, 16*(1), 7–33. https://doi.org/10.1002/icd.506

Volterra, V. (1926). Fluctuations in the abundance of a species considered mathematically. *Nature, 118*, 558–560. https://doi.org/10.1038/118558a0

von Davier, A. A., & Halpin, P. F. (2013). Collaborative problem-solving and the assessment of cognitive skills: Psychometric considerations. *ETS Research Report*.

Chapter 13
Social Networks Analysis

Mengxiao Zhu

Abstract Supported by advances in technology, simulation-, scenario- and game-based assessments (DiCerbo KE, Behrens JT, Implications of the digital ocean on current and future assessment. In: Lissitz R (eds) Computers and their impact on state assessment: recent history and predictions for the future. Information Age Publishing, Charlotte, pp 273–306, 2012; Mislevy RJ, Oranje A, Bauer MI, Von Davier A, Hao J, Corrigan S, . . . John M, Psychometric considerations in game-based assessment. GlassLab Report, 2014) provide opportunities for the students to interact with complex tasks. Rich process data can be collected during the assessment, such as log data of student response actions (e.g., Zhu M, Shu Z, von Davier AA, J Educ Measur 53(2):190–211, 2016b), keystroke data (e.g., Almond R, Deane P, Quinlan T, Wagner M, Sydorenko T, A preliminary analysis of keystroke log data from a timed writing task. In Educational testing service research report series (No. RR-12-23), pp. 1–61, 2012. https://doi.org/10.1002/j.2333-8504.2012.tb02305.x), and eye-tracking data (e.g., Tai RH, Loehr JF, Brigham FJ, Int J Res Method Educ 29(2):185–208, 2006. https://doi.org/10.1080/17437270600891614). Process data record the series of activities conducted by students during problem-solving processes and contain information not represented in the final answers. One useful direction in which to study process data is to explore how students transit from one action to other actions, or from one state to other states. In this chapter, we introduce the basic concepts and methods of Social Network Analysis (SNA) and discuss related applications in visualizing and analyzing process data using SNA to understand the transitions in process data.

The R or Python codes can be found at the GitHub repository of this book: https://github.com/jgbrainstorm/computational_psychometrics

This chapter was completed while the author was employed at Educational Testing Service.

M. Zhu (✉)
University of Science and Technology of China, Hefei, Anhui, China
e-mail: mxzhu@ustc.edu.cn

© Springer Nature Switzerland AG 2021

A. A. von Davier et al. (eds.), *Computational Psychometrics: New Methodologies for a New Generation of Digital Learning and Assessment*, Methodology of Educational Measurement and Assessment, https://doi.org/10.1007/978-3-030-74394-9_13

13.1 Overview of Social Networks Analysis

As an important research direction in modern sociology, Social Network Analysis (SNA) uses networks (also called graphs) to model humans and especially the connections among humans. In a social network, nodes are usually humans, and the links are various kinds of connections between humans, such as friendship (e.g., Goodreau et al., 2009), advice seeking (e.g., Cross et al., 2001), and trust (e.g., Burt & Knez, 1996). For the collection of data of social networks in a small group, participants are usually asked to complete questionnaires to report their social connections with all other members in the same group. To enable collection of samples of social network data from a big population, snowball sampling (Wasserman & Faust, 1994) is often adopted. In snowball sampling, a set of individuals are named as seed nodes and are asked to report a list of their contacts for the studied relationship. Further steps of snowball sampling can then be implemented by naming as seed nodes all of the individuals identified as contacts by the previous step seed nodes.

The modeling and analysis of social networks in sociology make use of graph theory (West, 2000), which uses graphs to mathematically model relations between objects as links and nodes. To capture structural features of social networks, measures and analysis techniques were developed in SNA, such as degree, density, centrality, and transitivity (Davis & Leinhardt, 1972; Wasserman & Faust, 1994, to be discussed in greater detail in Sect. 13.2). Many social network measures and features are deeply rooted in social theories. For example, to investigate the phenomenon of "small world", Stanley Milgram conducted the small-world experiment to find the distance between two random nodes in a social network (Milgram, 1967). Through a social experiment of mail relays, the results showed surprisingly short connections between two random humans in the United States. As another example, sociologists proposed and studied social capital (Burt et al., 2000), which measures an individual's social connections (e.g., degree and centrality) and connections among the focal individual's contacts (e.g., transitivity). Social capital is considered an important resource in organizations and social systems and can be used to explain the success of well-connected individuals and the importance of certain individuals bridging others with diverse backgrounds.

With the availability of vast amounts of network data from multiple research fields, recent decades witnessed significant expansion of network studies well beyond the social science domain. As a few examples, researchers studied metabolic networks of proteins, electric power grid networks, Internet networks of websites and users, and scientific collaboration networks of researchers (Newman, 2003). Related studies attracted researchers from a lot of fields, including biology, electrical engineering, computer science, physics, and many others. Measures and network models were proposed to analyze, assist, and understand the structures and dynamics of networks, especially large-scale networks. For instance, node degree distribution captures the distribution of the number of links connected to each node. Researchers found many networks in the real world with power-law node degree distributions called scale-free networks (Barabási & Albert, 1999), such as movie

actor collaboration networks, links to websites in the World Wide Web, and the power grid networks. These networks feature a few nodes with many links and many nodes with only a few connections. Researchers also developed models to capture the generative processes of networks, and the simplest generative network model is the random graph model (Erdős & Rényi, 1959). To generate a network with n nodes and m links, the process starts with a network with n nodes and no links, and two unconnected nodes are randomly chosen and connected, with this process repeated until all m links are created. The degree distributions of these networks follow the Poisson distribution, so this model is also called the Poisson random graph model.

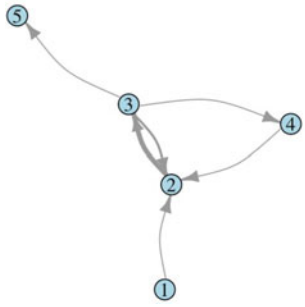
Social network analysis has also been widely applied in the education field to understand interactions among students or among students and teachers in classrooms (e.g., Grunspan et al., 2014), in distance learning environments (e.g., Haythornthwaite, 2001), and in the online forums of MOOC learning systems (Zhu et al., 2016a). Similar to applications in biology and computer science, network modeling and analysis do not have to be constrained to systems consisting of only humans as nodes. It can be expanded to model the learning and problem-solving processes consisting of knowledge, concepts, actions, and status as nodes. The links in networks can be used to model the relations, dependencies, and transitions among these nodes. For example, in responding to scenario-based tasks, students' problem-solving activities can be recorded in the log data. Network models can be used to model and to visualize such process data to recover and analyze the problem-solving processes (e.g., Zhu et al., 2016b). As another example, networks can be used to model students' transitions among different areas of interest (AOIs) with data collected using eyetracking devices (e.g., Zhu & Feng, 2015). More details on related studies will be provided in the following sections of this chapter.

13.2 Types of Networks and Basic Network Statistics

In this section, we provide the formal definition of a network and introduce different types of networks and some basic descriptive network statistics. A network consisting of a set of nodes and a set of links can be mathematically described in several ways. In this chapter, we adopt the so-called sociometric notation (Wasserman & Faust, 1994) and use the *adjacency matrix* to represent a network. We define a network $\mathcal{G}$ with a set of n nodes $V = \{v_1, v_2, .., v_n\}$ and m links $L = \{l_1, l_2, \ldots, l_m\}$. Using the adjacency matrix X, the set of n nodes can be represented by the rows and columns in the matrix. For a network with n nodes, the size of the adjacency matrix is $n \times n$. With the rows indicating the sending nodes and the columns indicating the receiving nodes, the links in $L = \{l_1, l_2, \ldots, l_m\}$ can be represented by the corresponding cells in the matrix. A link *from* node *i to* node *j* is indicated by the value of the (i, j)-th element x_{ij} of the adjacency matrix X, where $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, n$. Theoretically, a link can also be from a node to itself, called a self-loop, and can be represented by the values on the diagonal of the adjacency matrix.

For a network, if we distinguish the link *from* node *i to* node *j* and the link *from* node *j to* node *i*, this type of network is called a directed network. Some examples of directed networks include advice-seeking networks with links from the advisees to their advisors, paper citation networks with links from focal papers to papers cited by the focal papers, and email communication networks with links from the senders to the receivers. In contrast, if we do not distinguish the link *from* node *i to* node *j* and the link *from* node *j to* node *i*, and only consider links to exist *between* two nodes, then this type of network is called an undirected network. As a result, the adjacency matrix of an undirected network is symmetric. Some examples of undirected networks include friendship networks between students, co-authorship networks between authors, and train route networks between cities. Networks can also be weighted networks or unweighted networks. In unweighted networks, the links only represent the presence or absence of the relations; in weighted networks, the links represent not only the presence of the relations but also the strength of the relations. For example, for a friendship network, if it is weighted, the links can represent different levels of closeness between friends; if it is unweighted, the links can only represent whether or not two individuals are friends. The weights of the links can be reflected by the values of elements in the adjacency matrix. Unweighted networks can be considered as special cases of weighted networks with weights of all links being 1, and, consequently, their adjacency matrixes contain only 0's and 1's.

As a simple example, consider a student working on a set of five assessment items, Item 1 through Item 5. The student starts with the first item, may follow any order to answer the items, and can also revisit any previously finished items. A possible answering sequence of the items is {Item 1, Item 2, Item 3, Item 2, Item 3, Item 2, Item 3, Item 4, Item 2, Item 3, Item, 5}. To model this answering sequence, we can use a weighted and directed network, with nodes indicating items and links indicating transitions among items. The visualization of the transition network and its adjacency matrix is shown in Fig. 13.1. This network has five nodes, labeled as 1 through 5, and there are six directed links among these nodes. In the network plots, the thickness of the links indicates the frequencies of the transitions, and higher frequencies of transitions between links are represented by higher weights. For example, the link from node 2 to node 3 has the highest weight of 4, and the



	1	2	3	4	5
1	0	1	0	0	0
2	0	0	4	0	0
3	0	2	0	1	1
4	0	1	0	0	0
5	0	0	0	0	0

Fig. 13.1 An example of a simple weighted and directed network and its adjacency matrix

Table 13.1 Degrees of nodes in the example network

Node	Outdegree	Indegree	Weighted outdegree	Weighted indegree
Item 1	1	0	1	0
Item 2	1	3	4	4
Item 3	3	1	4	4
Item 4	1	1	1	1
Item 5	0	1	0	1

corresponding cell x_{23} of the adjacency matrix has the value 4, which showed that students moved from Item 2 to Item 3 four times. In the remainder of this section, we review several basic network statistics, especially those closely related to the analysis of process data. We provide both formal definitions and statistics using the above example.

- *Node Degree*. Node degree captures the number of links connected to a node. To calculate node degrees for a weighted network, the adjacency matrix X of needs to be dichotomized first: Values larger than 0 are replaced with 1's, and 0's remain as is. The node degree for node i is $d\left(v_i\right) = \sum_{j=1}^{n} x_{ij}$ for undirected networks. For a directed network, outdegree $d_O\left(v_i\right) = \sum_{j=1}^{n} x_{ij}$ captures the number of outgoing links, and indegree $d_I\left(v_i\right) = \sum_{j=1}^{n} x_{ji}$ captures the number of incoming links. For weighted networks, weighted degrees capture not only the number of links, but also the strength of the connections, which can be calculated similarly as above using the original (undichotomized) adjacency matrix. The degrees of the nodes in the network in Fig. 13.1 are shown in Table 13.1. In our example, Item 3 has the highest outdegree, Item 2 has the highest indegree, and they both have the highest weighted degrees. This shows that the student was struggling on these two items, and the student revisited Item 2 after working on several different items.

- *Network Density*. Links in a network may exist or be absent, and they may, likewise, convey distinctive weights. For a network, density captures the degree to which the nodes are interconnected with each other. For an undirected network with n nodes and m links, there can be a theoretical maximum of $\frac{n(n-1)}{2}$ links. The network density D is the total number of links m divided by the theoretical maximum: $D = \frac{2m}{n(n-1)}$. For a directed network with n nodes and m links, there can be a theoretical maximum of $n(n-1)$ links. The network density D is the total number of links m divided by the theoretical maximum: $D = \frac{m}{n(n-1)}$. For weighted networks, links need to be weighted by their strengths, so the original (undichotomized) adjacency matrix is used. The weighted density is $D_w = \frac{2\sum_{i=1}^{n}\sum_{j=i+1}^{n} x_{ij}}{n(n-1)}$ for an undirected network and $D_w = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij}}{n(n-1)}$ for a directed network. Higher density indicates more transitions among the nodes. The density of the example network is 0.3 indicating that 30% of all possible transition relations were present in the network, and the weighted density is 0.5

indicating that over any pairs of items, the probability to move from one item to another item is 0.5.

- *Network Reciprocity*. Reciprocity is defined on a directed network with link weight ignored and is calculated using features of pairs of nodes. A dyad is a pair of nodes and any links between them. If there is at least one link between these two nodes, it is called a non-null dyad. If both links exist in a dyad, it is called a mutual dyad (Wasserman & Faust, 1994, Chap. 13). Reciprocity is calculated as the number of mutual dyads divided by the total number of non-null dyads: $R = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij} * x_{ji}}{\sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij} \mid x_{ji}}$ (Butts, 2008a), where $x_{ij} \mid x_{ji} = 1$ if either x_{ij} or x_{ji} equals zero and $x_{ij} \mid x_{ji} = 0$ if both x_{ij} and x_{ji} are zero. Reciprocity captures the percentage of transitions that happened both ways. For the example network, among five non-null dyads, only one is reciprocal (between nodes 2 and 3) and the network reciprocity is 0.2, which shows that the student did not go back and forth between many items.
- *Network centralization*. The concept of centrality came from the social network, where some nodes are more active and better connected than others. One way to measure centrality is to use the degrees of nodes, and the resulting type of centrality is called degree centrality. For a connected network of n nodes (all nodes connected to at least one other node), the most connected node has $n - 1$ connections with all other nodes, and the least connected node has one connection. For a network, the node centralities usually vary across nodes. At one extreme, a star network has one central node and all other nodes having only one contact. At the other extreme, a circle network contains all nodes which have the same number of contacts and are each connected to two other nodes. To measure the network centralization, degree centralization (as in Freeman, 1979) for the network $\mathcal{G}$ is defined as $\mathrm{C} = \frac{\sum_{i=1}^{\mathrm{n}}\left[\max_{\mathrm{j}\in\{1,2,\ldots,\mathrm{n}\}} d(v_j) - \mathrm{d}(v_i)\right]}{\max \sum_{i=1}^{\mathrm{n}}\left[\max_{\mathrm{j}\in\{1,2,\ldots,\mathrm{n}\}} d(v_j) - \mathrm{d}(v_i)\right]}$, where $d(v_i)$ and $d(v_j)$ are the degrees of node i and node j, respectively. In this definition, the numerator captures the overall difference between each node's degree and the highest node degree in the network, and the denominator is the theoretical maximum overall difference calculated on all possible networks with n nodes. The largest overall differences of node degrees happen in a star network, and this maximum is used to calculate the denominator. Thus, for a star network, $C = 1$, and for a circle network, $C = 0$. For a directed network, two centralization measures are defined using either outdegree C_O or indegree C_I. For our example network, $C_O = C_I = 0.56$. The value is in the midway between 0 and 1, indicating that the node degrees are not well balanced across nodes.
- *Triad census*. To capture the local patterns of three nodes and links among these nodes in a directed network, researchers (Davis & Leinhardt, 1972; Holland & Leinhardt, 1970) proposed the triad census, which enumerated all 16 possible isomorphic patterns and names using numbers and letters (as shown in Fig. 13.2). For the names of each pattern, the first three numbers indicate the number of mutual dyads, asymmetric dyads, and null dyads. There may be a letter after the number, indicating the shape of the triad, down (D), up (U), cyclic (C), or

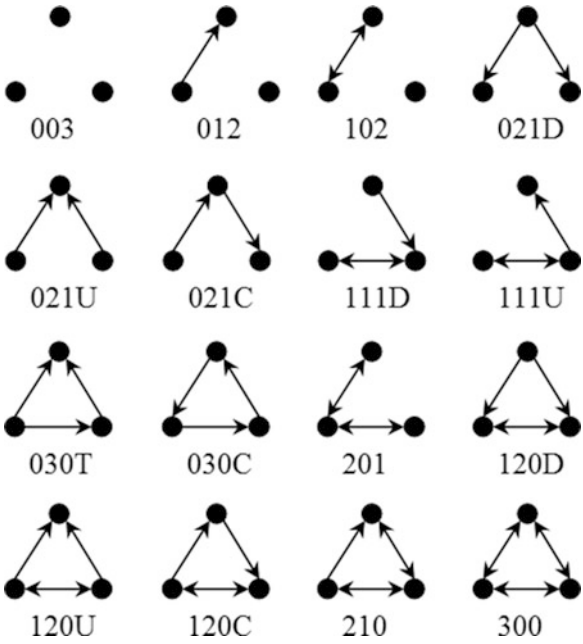


Fig. 13.2 Triad census

transitive (T). Triad census counts the number of triadic patterns in a network and enables the investigation of local patterns among three nodes. In our sample network, seven out of 16 triad patterns exist, with four 012, and one 003, 021D, 021U, 111D, 111U, and 120C.

13.3 Examples of SNA in Educational Measurement

In this section, we discuss two studies that applied SNA and network analysis in general to analyzing process data. Each of the two studies used process data from different sources and answered different research questions, but they shared similar network analysis approaches. We hope discussions of these studies can demonstrate the application of SNA in analyzing process data and will inspire new applications.

13.3.1 Study 1: Response Sequences

As part of the NAEP Technology and Engineering Literacy (TEL) assessment developed by the National Center for Education Statistics, students were asked to work on a scenario-based task with the goal of fixing a malfunctioned pump

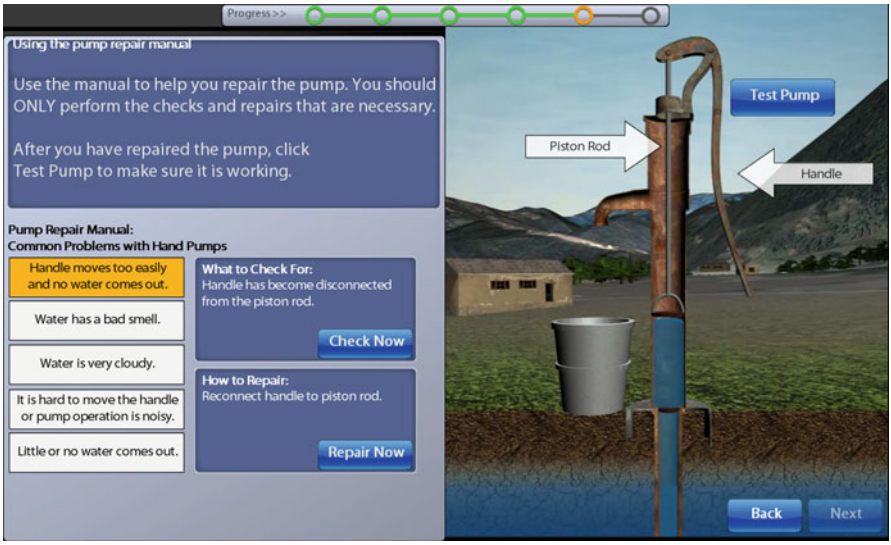


Fig. 13.3 Interface of the wells task

(http://nces.ed.gov/nationsreportcard/tel/wells_item.aspx). This task requires the students to solve the problem by applying their TEL knowledge and skills. In the interactive interface, students are provided with five common problems that a pump may have. For each potential problem, there is also a "Check Now" action (C) that will check to see if the pump has this problem, and a "Repair Now" action (R) that will perform repairs to address the potential problem. There is also a "Test Pump" action (P) that enables the student to test the pump to see if it is working properly. A screen shot of the task is shown in Fig. 13.3. In this task, the well has the fourth and the fifth problems in the manual, and the students should have this information from the scaffolding in the previous scene. The well can be fixed if students check and repair the fourth and the fifth problems. If students only conduct necessary actions of checking and repairing the fourth and the fifth problems and test the pump, they score higher on problem-solving efficiency. If students also conduct unnecessary actions of checking and repairing other problems, they score lower. Students have to fix the well before they can move on to the next task.

To model students' problem-solving processes in terms of what actions students choose to take and how they transition from one action to another, the nodes in the transition networks are the actions, and the links are the directed transitions from one set of actions to another set. To indicate the beginning and the end of the sequences, two more nodes "start" and "end" were added to the networks. Figure 13.4 shows the visualization of a sample action sequence (C4,P,P,R4,P,C5,R5,P) and the overall transition network of all students (more details of this study are available in Zhu et al., 2016b). A transition network for multiple students can be constructed by summing up their individual transition networks. In the overall transition network,

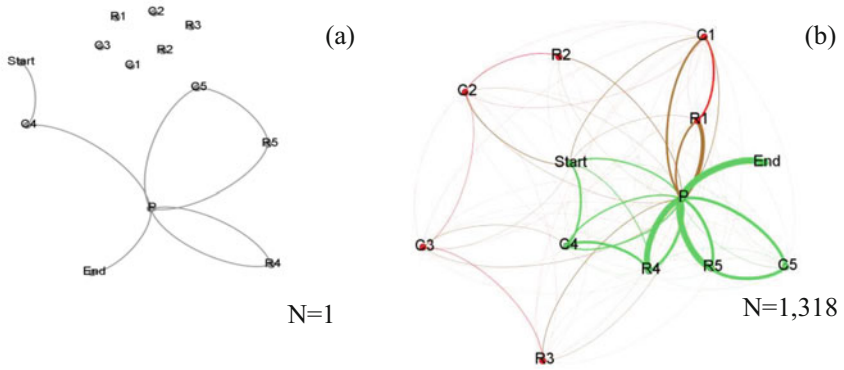


Fig. 13.4 Transition networks (**a**) Sample network from one student, (**b**) Transition network from a sample of 1318 students. (Reprinted from "Using Networks to Visualize and Analyze Process Data for Educational Assessment" by Zhu et al. 2016b)

green links indicate necessary actions to solve the problem, red links indicate unnecessary actions, and the width of the links indicates the frequency of the transitions. Observations reveal patterns in student actions. For example, it can be seen from the overall transition network that from the "start" of the sequences "C4" (check for the fourth problem) was one of the most frequently chosen actions, which is also a necessary action. The other necessary checking action "C5" (check for the fifth problem) did not have equal popularity, even though they are both necessary actions. This observed pattern might be related to the fact that the fourth problem appeared above the fifth problem in the provided repair manual, and students may have just conveniently selected the first reasonable action.

Further analysis on the network measures also showed that the transition patterns were different for students who obtained high efficiency scores and for those who obtained low efficiency scores. For example, the weighted density was higher for students with lower efficiency scores and lower for students with higher scores. This can be explained by the fact that students with lower efficiency scores conducted more unnecessary actions in solving the problem and, thus, had more transitions among actions, which was reflected by higher weighted density. Weighted transition network density was also found to have significant predicting power on students' efficiency scores. Another interesting finding was that students who received medium scores had the highest reciprocity, and students with very high or very low scores had low reciprocity. With reciprocity being an indicator of the percentage of transitions back and forth between nodes, higher reciprocity indicates more repeated actions. Low efficiency students may not have known what to do and, thus, explored all possible actions, which resulted in fewer repeated steps. High efficiency students who knew what to do went directly to the correct actions and also did not go back and forth between actions. However, students in the middle struggled and repeated more actions, which resulted in a lot of reciprocal links. Using transition networks to model action sequences, we can discover many interesting patterns and can compare

features of students in different subgroups. However, while transition networks truthfully capture the transitions among two adjacent actions, they omit sequential information involving more than two actions.

13.3.2 Study 2: Eye Movements During Problem Solving

As a second example, we discuss how networks were used to model eye movements during students' problem solving processes. In an exploratory study (Zhu & Feng, 2015), eyetracking data were collected from 37 students while they were working on the Moving Sidewalks task, a math task developed as part of the Cognitively Based Assessment *of*, *for*, and *as* Learning (*CBAL*®) initiative (Bennett, 2010). In this task, students were provided with a plot of the linear relationship between time and the distance travelled on the moving sidewalk and were asked to answer several questions using information from the plot. Eye gazes are widely accepted by researchers as being indicators of people's attention focuses (Findlay & Walker, 1999; Posner et al., 1980).

Eyetracking data are usually collected by periodically scanning where the participants are gazing at on a screen. The resulting data include the time stamps and the locations on the screen. One natural way to model and to visualize the eyetracking data is to use scan path (Holmqvist et al., 2011), which lays out all locations and connections between two adjacent locations. Figure 13.5 shows the scan path of a sample student while working on the Moving Sidewalks task. The

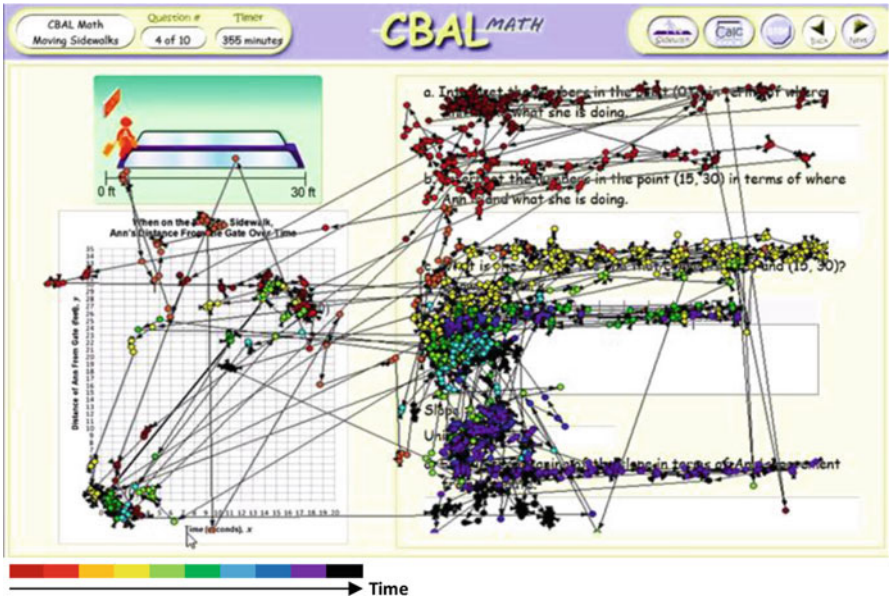


Fig. 13.5 A sample scan path from a sample student

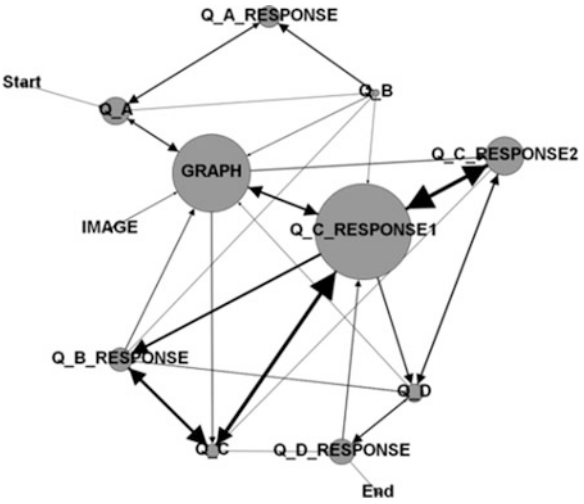


Fig. 13.6 AOI transition network for a sample student

nodes were color coded, with colors on the left side of the spectrum indicating earlier time during problem solving and colors on the right side indicating later time (see the bottom left of Fig. 13.5). It can be seen that in this sample scan path, this student started with the first item and roughly followed the order of the items. The student also checked the plot on the left-hand side of the screen while answering some items.

The scan path constructed from the raw eyetracking data includes too much information and makes analyzing such a network and especially interpreting the results difficult. One way to aggregate the scan path data is to use Areas of Interest (AOIs; Holmqvist et al., 2011), which divide the screen into several non-overlapping areas. By mapping eyetracking location information into the limited AOIs, transition networks can be constructed for each student. An example of an AOI transition network is shown in Fig. 13.6. In this example, the size of the nodes indicates the amount of time spent at the AOIs. The links are directed and weighted, and the weights demonstrates frequencies of transitions. A lot of different transition patterns can be observed in the transition network. For the eyetracking data, it makes a lot of sense to study the detailed local patterns in the transition networks. For all 37 students, it was found that they all have many reciprocal links on average. The mean of the network reciprocity was $R = 0.61$; that is, on average 61% of the links were reciprocal. Students with high and low performance also showed different triadic structural features. For example, low-performing students had more 003 and 021D triad patterns than high-performing students but fewer 111U triad patterns. The 003 structure has three isolated nodes, 021D is the structure with one node with two outgoing links pointed to two other nodes, and 111U is the structure of reciprocal links between two nodes and a link pointing to the third node. Across these different local patterns, it can be seen that low-performing students made fewer connections among AOIs (003 being observed more frequently) and were

less strategic by moving away from one AOI to two unrelated AOIs (021D being observed more frequently). High-performing students, on the other hand, were more strategic and had back and forth moves between two AOIs before branching out to a third AOI.

13.4 Data Example in R Programming Language

In this section, we provide the R code to create and analyze the sample network introduced in Sect. 13.2. Two R packages were used. The *igraph* package (Csárdi & Nepusz, 2006) was used to plot the network, and the *sna* package (Butts, 2008b) was used to calculate the network statistics.

```
#### Create the adjacency matrix for the example network
net.matrix <- matrix(c(0,1,0,0,0, 0,0,4,0,0, 0,2,0,1,1,
   0,1,0,0,0, 0,0,0,0,0),
                  nrow = 5, ncol = 5, byrow = TRUE,
                  dimnames = list(c("1", "2", "3", "4", "5"),
                                  c("1", "2", "3", "4", "5")))

#### Network Visualization
library(igraph)
net=graph.adjacency(net.matrix,mode="directed",weighted=TRUE,
     diag=FALSE)
l <- layout.fruchterman.reingold(net)
plot.igraph(net,layout=l, vertex.label.cex=1,
     vertex.label.color="black",
     edge.color="gray60",vertex.color="light blue",
     edge.width=E(net)$weight,
     edge.arrow.size=0.4,edge.curved=.3)

#### Network Analysis
library(sna)

## Create the network object
net <- network(net.matrix,matrix.type="adjacency",directed=TRUE)

## Network Size
network.size(net)

## Outdegrees and Indegrees for Nodes
degree(net,cmode="outdegree")
degree(net,cmode="indegree")

## Weighted Outdegrees and Indegrees for Nodes
degree(net.matrix,cmode="outdegree")
degree(net.matrix,cmode="indegree")

## Network Density
gden(net)

## Weighted Density
gden(net.matrix)
```

```
## Network Reciprocity
grecip(net,measure="dyadic.nonnull")

## Network Centralization
centralization(net,degree,cmode="outdegree")
centralization(net,degree,cmode="indegree")

## Triad Census
triad.census(net, mode = "digraph")
```

13.5 Summary

This chapter introduced the method of social network analysis (SNA) and discussed its application in analyzing process data collected in digitally delivered interactive assessments. Section 13.1 reviewed the background and development of SNA as an interdisciplinary method. Section 13.2 covered the basics of SNA. It included the definitions of the different types of networks and the formal definition and representation of networks. It also introduced the definitions and formulas for calculating several network statistics using a simple toy network as an example. In Sect. 13.3, two empirical studies using SNA to analyze process data were reviewed. The first study focused on modeling the response sequences in the process data in a scenario-based task, and the second study modeled students' eye-movement patterns during problem solving using networks. Section 13.4 included the R code to create the example network in Sect. 13.2, the visualization of the network, and the calculations of basic network statistics introduced in this chapter using related R packages. Even though the reviews of SNA in this chapter are brief and far from complete or comprehensive, we hope it can prepare interested readers with the basic knowledge necessary to get started with SNA and can inspire innovative applications of SNA in modeling and analyzing process data.

References

Almond, R., Deane, P., Quinlan, T., Wagner, M., & Sydorenko, T. (2012). A preliminary analysis of keystroke log data from a timed writing task. In *Educational testing service research report series* (No. RR-12-23) (pp. 1–61). https://doi.org/10.1002/j.2333-8504.2012.tb02305.x

Barabási, A.-L., & Albert, R. (1999). Emergence of scaling in random networks. *Science, 286*(5439), 509–512. https://doi.org/10.1126/science.286.5439.509

Bennett, R. E. (2010). Cognitively based assessment of, for, and as learning (CBAL): A preliminary theory of action for summative and formative assessment. *Measurement: Interdisciplinary Research & Perspectives, 8*(2), 70–91.

Burt, R. S., Hogarth, R. M., & Michaud, C. (2000). The social capital of French and American managers. *Organization Science, 11*(2), 123–147. https://doi.org/10.1287/orsc.11.2.123.12506

Burt, R. S., & Knez, M. (1996). Trust and third-party gossip. In R. M. Kramer & T. R. Tyler (Eds.), *Trust in organizations: Frontiers of theory and research* (pp. 68–69). Sage.

Butts, C. T. (2008a). Social network analysis: A methodological introduction. *Asian Journal of Social Psychology, 11*(1), 13–41. https://doi.org/10.1111/j.1467-839X.2007.00241.x

Butts, C. T. (2008b). Social network analysis with sna. *Journal of Statistical Software, 24*(6).

Cross, R., Borgatti, S. P., & Parker, A. (2001). Beyond answers: Dimensions of the advice network. *Social Networks, 23*(3), 215–235.

Csárdi, G., & Nepusz, T. (2006). The igraph software package for complex network research. *Complex Systems, 1695*(5), 1–9. https://doi.org/10.3724/SP.J.1087.2009.02191

Davis, J. A., & Leinhardt, S. (1972). The structure of positive interpersonal relations in small groups. In J. Berger (Ed.), *Sociological theories in progress* (Vol. 2, pp. 218–251). Houghton Mifflin.

DiCerbo, K. E., & Behrens, J. T. (2012). Implications of the digital ocean on current and future assessment. In R. Lissitz (Ed.), *Computers and their impact on state assessment: Recent history and predictions for the future* (pp. 273–306). Information Age Publishing.

Erdős, P., & Rényi, A. (1959). On random graphs I. *Publicationes Mathematicae, 6*, 290–297. https://doi.org/10.2307/1999405

Findlay, J. M., & Walker, R. (1999). A model of saccade generation based on parallel processing and competitive inhibition. *Behavioral & Brain Sciences, 22*(4), 661–721.

Freeman, L. (1979). Centrality in social networks I: Conceptual clarification. *Social Networks, 1*, 215–239.

Goodreau, S. M., Kitts, J. A., & Morris, M. (2009). Birds of a feather, or friend of a friend? Using exponential random graph models to investigate adolescent social networks. *Demography, 46*(1), 103–125. https://doi.org/10.1353/dem.0.0045

Grunspan, D. Z., Wiggins, B. L., & Goodreau, S. M. (2014). Understanding classrooms through social network analysis: A primer for social network analysis in education research. *Cell Biology Education, 13*(2), 167–178. https://doi.org/10.1187/cbe.13-08-0162

Haythornthwaite, C. (2001). Exploring multiplexity: Social network structures in a computer supported distance learning class. *The Information Society, 17*(3), 211–226.

Holland, P. W., & Leinhardt, S. (1970). A method for detecting structure in sociometric data. *American Journal of Sociology, 76*(3), 492–513.

Holmqvist, K., Nyström, M., & Andersson, R. (2011). *Eye tracking: A comprehensive guide to methods and measures*. Oxford University Press.

Milgram, S. (1967). The small world problem. *Psychology Today, 2*(1), 60–67.

Mislevy, R. J., Oranje, A., Bauer, M. I., Von Davier, A., Hao, J., Corrigan, S., … John, M. (2014). Psychometric considerations in game-based assessment. *GlassLab Report*.

Newman, M. E. J. (2003). The structure and function of complex networks. *SIAM Review, 45*, 167–256.

Posner, M. I., Snyder, C. R., & Davidson, B. J. (1980). Attention and the detection of signals. *Journal of Experimental Psychology: General, 109*(2), 160–174.

Tai, R. H., Loehr, J. F., & Brigham, F. J. (2006). An exploration of the use of eye-gaze tracking to study problem-solving on standardized science assessments. *International Journal of Research & Method in Education, 29*(2), 185–208. https://doi.org/10.1080/17437270600891614

Wasserman, S., & Faust, K. (1994). *Social network analysis: Methods and applications*. Cambridge University Press.

West, D. B. (2000). *Introduction to graph theory* (2nd ed.). Prentice Hall.

Zhu, M., Bergner, Y., Zhang, Y., Baker, R., Wang, Y., Paquette, L., & Barnes, T. (2016a). Longitudinal engagement, performance, and social connectivity: A MOOC case study using exponential random graph models. In *Proceedings of the 6th international learning analytics and knowledge conference (LAK '16)*.

Zhu, M., & Feng, G. (2015). An exploratory study using social network analysis to model eye movements in mathematics problem solving. In *Proceedings of the 5th international learning analytics and knowledge conference (LAK '15)* (pp. 383–387). ACM. https://doi.org/10.1145/2723576.2723591

Zhu, M., Shu, Z., & von Davier, A. A. (2016b). Using networks to visualize and analyze process data for educational assessment. *Journal of Educational Measurement, 53*(2), 190–211.

Chapter 14
Text Mining and Automated Scoring

Michael Flor and Jiangang Hao

Abstract Natural Language Processing (NLP) is playing an increasingly important role in learning and assessments. Some typical applications of NLP in education include automated scoring, automated item generation, conversation-based assessments, writing assistants, text mining for education, and so on. In this chapter, we aim at introducing some basics of NLP through two typical applications in educational contexts, text mining and automated scoring. We hope readers can get an overall picture of NLP and get familiarized with some basic tools for handling natural language data, which may serve as stepping stones for their future work with NLP.

14.1 Overview

Natural Language Processing, also known as Computational Linguistics, is an interdisciplinary area of research, spanning computer science and artificial intelligence (AI), linguistics and cognitive psychology, as well as other disciplines, such as mathematics, logic, philosophy, and neuroscience. The central questions of research revolve around the notion of enabling computer programs to automatically analyze and represent natural human language (e.g. English). Sub-areas of inquiry include natural language understanding and natural language generation (for text-based data), and speech recognition and generation (for spoken data). As an area of study, computational linguistics has both theoretical and applied perspectives, and the notion of NLP often refers to the more applied areas of the field.

The R or Python codes can be found at the GitHub repository of this book: https://github.com/jgbrainstorm/computational_psychometrics

M. Flor (✉) · J. Hao
Educational Testing Service, Princeton, NJ, USA
e-mail: mflor@ets.org; jhao@ets.org

© Springer Nature Switzerland AG 2021

A. A. von Davier et al. (eds.), *Computational Psychometrics: New Methodologies for a New Generation of Digital Learning and Assessment*, Methodology of Educational Measurement and Assessment, https://doi.org/10.1007/978-3-030-74394-9_14

Research on computer-based processing of natural language covers a diverse range of language-related activities. Traditional areas of research include text document analysis, extraction and accumulation of information, facilitation of information search activities and assisting in correspondence, such as translation and dictation. Modern applications of such technologies are web-search engines and information extraction, news summarization, spam-filtering, and voice recognition (including for services on phones and for home devices).

While computational linguistics has its roots in linguistics, logic and philosophy, early research has crystallized in the 1950s, with a concerted effort to develop computer programs for automatic translation of scientific journal articles from Russian to English (Slocum, 1985). Thus, machine translation is one of the oldest application areas of NLP (Hutchins, 1995). It gave rise to a new field of study that focuses on developing algorithms and software for processing language data for a variety of purposes.

It is common to consider that the early NLP approaches were mostly oriented toward hand-crafted sets of rules directly coded into computer programs for handling language inputs. By the late 1980s and early 1990s, statistical approaches became dominant in NLP. In particular, machine learning approaches enable rules and regularities in the data to be learned by applying learning algorithms over large quantities of data (i.e. speech and text corpora). Statistical approaches brought two major advancements to the field (Basili & Magnini, 2012). One of them was the increased robustness in the face of noisy data. Rule-based systems would often break down when required to handle noisy or incomplete data. Statistical models which use probabilistic decisions, have shown to be able to handle noisy and difficult data at least somewhat successfully, with graceful degradation rather than total failure. The other advantage is that learning regularities from data (often called language modeling) is a promising approach for overcoming the knowledge acquisition bottleneck. Developing very complex rule-based systems requires large coordinated efforts of many human experts (which is also extremely costly), whereas statistical approaches can often learn from data in ways that are faster, less expensive, and often more accurate.

For historical accuracy, it should be stressed that statistical approaches were present from the early days of computational linguistics; for example, a statistical approach was used in the early 1960s in determining the authorship of the Federalist Papers (Mosteller & Wallace, 1963). On the other hand, Chiticariu, Li, and Reiss (2013) demonstrate that rule-based systems are still widely popular for industrial-scale applications in some areas, such as information extraction.

Statistical methods in NLP have also evolved over the years. Early approaches popularized the use of classic statistical methods, such as Bayesian classification, logistic regression, linear regression, etc. With the advancement of statistical approaches in other areas of research, NLP embraced a variety of machine learning methods, such as Support Vector Machines (SVM), decision trees, random forests, etc. (Jurafsky & Martin, 2008; Manning et al., 1999). In recent years NLP has also embraced another veteran paradigm: neural networks, a family of machine-learning architectures that are vaguely inspired by inter-connectivity of neurons in biological

nervous systems. This new incarnation, often called Deep Learning, places a strong emphasis on developing systems that can learn a language processing task from a multitude of examples rather than being specifically constructed/programmed for a task (Li, 2017; Manning, 2015).

An important distinction in NLP is that between NLP-intrinsic tasks and extrinsic NLP tasks, systems and applications. The utility of extrinsic tasks is easier to understand – those are NLP tasks where the output has some direct value to a significant community of potential users. A classic direct extrinsic task is machine translation. The value of good machine translation is obvious – it is directly usable, fast and reduces the cost of laborious human effort. Another extrinsic task is text classification. Text classification makes it possible to assign predefined categories to a document. This in turn, allows organizing sets of documents (or even just text snippets) in a variety of ways, and subsequently performing actions based on such classification. Spam filtering of email is a widely known service that relies on a text-classification process. Other examples of classic NLP tasks with direct-to-user value are spelling and grammar correction, voice recognition and speech synthesis. A more modern NLP task is question-answering, in which an NLP system delivers actual answers and not just relevant documents (Kolomiyets & Moens, 2011). A combination of tasks is becoming more and more popular, for example applications such as Siri and Google Assistant involve advanced voice recognition, question answering, and speech synthesis. A relatively recent popular application of NLP is sentiment analysis, which aims to determine the opinions and attitudes of writers/speakers, or at least contextual polarity (good vs. bad, pro vs. con), towards some topic or event, such as a commercial product or political stance, by analyzing language inputs, such as customer reviews or polls and tweets (Feldman, 2013; Pang & Lee, 2008).

Much less well known to the general public are the intrinsic NLP tasks. Those are tasks that involve analysis of the language inputs, but the outputs are meaningful mostly just for the specialists. Some of those tasks involve automatic detection of important linguistic structures. These tasks include word lemmatization – conversion of words to their base form/citation form), and general morphological processing (recognition of inflected and conjugated words, recognition of derivational relations, e.g. employed – employee), part-of-speech tagging (recognition of words as nouns, verbs, adjectives, etc., in potentially ambiguous inputs), named entity recognition (e.g. names of people, organizations, places and locations, etc.), grammar parsing (recognition of phrases and sentence structures). Another intrinsic NLP task is semantic role labelling. This refers to recovering of abstract sematic relations between predicates (typically verbs) and various role-fillers in a sentence (essentially determining who did what to whom, when, and where). Intrinsic NLP tasks are often very useful for real-world applications, being either essential building blocks or enabling substantial improvements for the latter.

Many NLP tasks have a status that is between fundamental intrinsic and real-word application. Consider dialogic systems, ranging from chatbots to sophisticated customer-interface systems. Such systems need to convert the input (spoken or typed) into a machine-readable format, detect user intent and any entities mentioned,

and then generate responses, while keeping track of new information and changes in the discourse (Tur & De Mori, 2011; Wang, 2005). The final system would then be tuned for a specific application, such as handling an intelligent conversation with the user (e.g. a website customer) or assisting tourists at an information kiosk.

In the subsequent sections, we will focus on two specific areas of NLP that are very relevant to applications in an educational context – text mining and automated scoring. We begin with text-mining, which is not a self-contained task – it depends and adapts to a variety of user goals. We use it as an illustration of some basic NLP steps. Next, we describe automated essay scoring, which is an example of a complete educational application of NLP.

14.2 Text Mining

Text mining, also known as text data mining, refers to the discovery of patterns and meaningful information from texts through means such as pattern recognition and machine learning (Wikipedia, 2019). Typical text mining tasks can include sentiment analysis, topic analysis, text categorization, document summarization, text clustering, entity extraction and the mining of the relationship among the entities. The details of each of these tasks are easily worthy of a whole volume of description, and are beyond the scope of this chapter. In what follows, we try to give readers a general idea of how text mining is conducted in practice through some simple but typical examples. To proceed, we assume readers are already familiar with the Python programming language and Jupyter Notebook. For readers who want to know more about them, please refer to Chap. 8 and references therein.

14.2.1 Text Preprocessing

A prerequisite step to manipulating natural language mathematically or statistically is to convert the raw texts[1] into numerical representations. These numerical representations, often called features, are developed through a process called feature engineering (also *feature extraction*). In principle, there could be an infinite number of ways to create feature representations from the same texts for different purposes. For example, some features can be more lexicalized, depending entirely on the specific words or phrases. A typical example of such features is known as "bag of words," where the numerical representation is simply the set of counts of different

[1]The term 'raw text' refers to the form in which a text appears in the 'real-world' electronic application (e.g. a word-processor file, an electronic file of a student's essay, a text from a webpage, etc.). Usually, it is presumed that the extra markup is already stripped off (e.g. HTML or XML markup), so 'raw text' often means just the long string of the text characters.

words/phrases appearing in the texts. On the other hand, some other features can be less lexicalized, depending more on the relationship among words/phrases rather than the specific words/phrases themselves. For example, grammars are sets of rules that define the relationship among (classes of) words and phrases. The frequency of different grammar constructions (e.g. noun phrases) could be a numerical representation along this line. In practice, the application purpose of a given text mining task usually guides the choice of the set of feature representations.

Most raw texts in the real world are not ready for immediate feature engineering, and a preprocessing procedure is needed. The preprocessing procedure typically involves two steps. The first step is known as tokenization, through which the raw texts are chunked into a list of basic units of analysis. These units are called tokens (usually words and phrases, punctuation, numbers, etc.). The second step is a cleaning procedure,[2] which removes and/or normalizes the tokens. Typical examples of this step include case normalization, stop word[3] removal, and lemmatization. In Fig. 14.1, we show some example implementations of the tokenization and cleaning in Python. After the preprocessing, the texts are ready for further feature engineering.

14.2.2 N-Gram and Document-Term Matrix

After the preprocessing, one needs to represent the texts by numerical features so that statistical and mathematical manipulation of the texts are possible. As mentioned above, there are many ways to represent the texts numerically, depending on the purpose of the analysis. The simplest and most straightforward one is probably the so-called n-gram representation.[4] The n-gram representation is essentially the frequency of the single words (uni-gram, also known as the bag of words), two consecutive words (bi-gram), and n consecutive words (n-gram). The n-gram representation is a simple yet effective way of representing texts and works well for many practical applications, such as text classification and topic analysis.

The n-gram representation is powerful, but also has limitations. If one considers the unigram case, the relationship among the words is not adequately captured. To circumvent this, higher-order grams, such as bigram, trigram, and so on, are needed. However, including higher-order grams inevitably makes the numerical matrix very sparse, which causes problems when manipulating it mathematically. Word

[2] Text 'cleaning' is a rather subjective notion – which components in a text should be discarded and which should be retained depends largely on the particular application. There are some common trends, though, as illustrated in this section. For example, in many cases punctuation might be discarded.

[3] 'Stop words' are words that are eliminated (filtered out) from a document before further processing. In most cases those are very common words in a language (like *the, an, of, out*), which are not considered to contribute much to distinguishing the content of a text.

[4] Note that n-gram is often discussed in the context of language models.

```
In [1]: from nltk.tokenize import word_tokenize, sent_tokenize
        from nltk.corpus import stopwords
        from nltk.stem.porter import PorterStemmer
        from nltk import pos_tag
        from spellchecker import SpellChecker
        import string

In [2]: text = 'The class is over. I hopep it is intersting to you. Pl
        ease let me knoww if not.'

In [3]: #change to lower case
        text.lower()

Out[3]: 'the class is over. i hopep it is intersting to you. please let
        me knoww if not.'

In [4]: # word tokenization
        word_tokens = word_tokenize(text)
        print(word_tokens)

        ['The', 'class', 'is', 'over', '.', 'I', 'hopep', 'it', 'is',
        'intersting', 'to', 'you', '.', 'Please', 'let', 'me', 'knoww',
        'if', 'not', '.']

In [5]: # remove stop words and punctuations
        stopword_list = stopwords.words('english')
        punctuation_list = list(string.punctuation)
        cleaned_text = [txt for txt in word_tokenize(text.lower()) if
        txt not in stopword_list+punctuation_list]
        print(cleaned_text)

        ['class', 'hopep', 'intersting', 'please', 'let', 'knoww']

In [6]: # typo correction
        spell = SpellChecker()
        corrected_text = [spell.correction(wd) for wd in cleaned_text]
        print(corrected_text)

        ['class', 'hope', 'interesting', 'please', 'let', 'know']

In [7]: # part of speech tagging
        pos_tag(corrected_text)

Out[7]: [('class', 'NN'),
         ('hope', 'NN'),
         ('interesting', 'VBG'),
         ('please', 'JJ'),
         ('let', 'NN'),
         ('know', 'VB')]

In [8]: # Stemming the words
        porter = PorterStemmer()
        stem_words = [porter.stem(txt) for txt in corrected_text]
        list(zip(corrected_text,stem_words))

Out[8]: [('class', 'class'),
         ('hope', 'hope'),
         ('interesting', 'interest'),
         ('please', 'pleas'),
         ('let', 'let'),
         ('know', 'know')]
```

Fig. 14.1 Examples of text preprocessing in Python. The comment lines in the cells explain the main purpose of the corresponding step. Note that the packages and methods used in the examples are motivated by pedagogical clarity and thus may not necessarily be the most efficient computationally

embeddings are another way to represent words as numerical vectors. This technique comes from a linguistic hypothesis known as the Distributional hypothesis. It states that words appearing in the same contexts generally have similar meanings (Harris, 1954). With word embeddings, a neural network model is used to model the relationship among the words based on a large training corpus and represent each word (token) as a numerical vector. Depending on the specific modeling scheme, there are different ways to instantiate the embedding. For example, the Word2Vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014) are two popular word embedding instantiations. For more details on such embeddings see Levy and Goldberg (2014) and Mikolov et al. (2018). It is worth noting that the word vectors under the Word2Vec and GloVe do not address the polysemy issue, e.g., a word may have different meanings in different contexts. More recently, new embedding approaches that consider the word contexts have been developed. Typical examples include the Embeddings from Language Models (ELMO; Peters et al., 2018) and Bidirectional Encoder Representations from Transformers (BERT; Devlin et al., 2018).

In practical applications, one often deals with multiple documents rather than one. Here, the document refers to a text that is considered to be an independent unit of analysis. As such, a document can be a sentence, a paragraph, or a book, depending on the intended granularity of a given analysis. If we represent a single document as a numerical vector of certain features, then for multiple documents, we can organize the vectors into a matrix. In the special case that we consider only the unigram features, such a matrix is called "document-term" matrix, with its rows indexing the documents and the columns indexing the unique words. The values in the cells of the document-term matrix are the counts of the terms appearing in the corresponding documents. Figure 14.2 shows an example of the document-term matrix.

The document-term matrix is a special case of the feature-observation matrix widely used in machine learning. Based on such a matrix, one can easily perform supervised machine learning tasks, such as document classification, and unsupervised machine learning tasks, such as document clustering. The raw counts in a

	Term 1	*Term 2*	...	*Term t*
Document 1	2	1	...	...
Document 2	5	1	...	...
Document 3	6	1	...	...
...	...	...	...	...
Document d	1	2	...	...

Fig. 14.2 Document-term matrix

document-term matrix have some clear limitations, considering that the frequency of different terms depends on the length of the documents and certain words have higher baseline frequency than other words. Therefore, a weighting scheme needs to be introduced to adjust the raw counts of the appearance of different terms in different documents. The most popular one is known as TF-IDF weighting. The TF here means term frequency (count of word instances in a document) and the IDF refers to inverse document frequency (in how many documents a word appears). The TF part accounts for the varied length of the documents while the IDF part accounts for the baseline frequency of different words in all documents. To apply the TF weight to $term_j$, one will do the following:

$$TF\left(\text{term}_\text{j}\right) = \frac{\text{Total frequency of term}_\text{j} \text{ in document}_\text{i}}{\text{Total frequency of all terms in document}_\text{i}}$$

To apply the IDF weight to $term_j$, one will perform

$$IDF\left(\text{term}_\text{j}\right) = functionof\left(\frac{\text{Total number of documents}}{\text{Number of documents containing term}_\text{j}}\right)$$

The most common form for computing IDF is

$$IDF\left(\text{term}_\text{j}\right) = \log\frac{\text{Total number of documents}}{1 + \text{Number of documents containing term}_\text{j}}$$

By combining the two, a TF-IDF weight for $term_j$, will be

$$TF \bullet IDF\left(\text{term}_\text{j}\right) = TF\left(\text{term}_\text{j}\right) \times IDF\left(\text{term}_\text{j}\right)$$

In practice, there are other variants of the TF-IDF weighting scheme, though they all try to account for the effects of varied document length and varied baseline frequency of different terms. A TF-IDF weighted/transformed document-term matrix is often the actual input of different text mining tasks. Figure 14.3 shows how to turn a list of three sentences (documents) into a document-term matrix with unigram representation for each sentence and how to apply TF-IDF weights in Python.

14.2.3 Topic Modeling

With a set of documents, two of the typical questions one wants to ask are what topics are covered by these documents and how these documents can be grouped based on the topics. Topic modeling provides a set of methodologies to answer these questions. There are two popular algorithms, Latent Semantic Analysis (LSA; Deerwester et al., 1990) and Latent Dirichlet Allocation (LDA; Blei et al., 2003).

```
In [9]: # ngram representation
        from sklearn.feature_extraction.text import CountVectorizer, T
        fidfVectorizer
        import pandas as pd
```

```
In [10]: # sentence tokenization
         sentence_list = sent_tokenize(text)
         print(sentence_list)
```

```
['The class is over.', 'I hopep it is intersting to you.', 'Ple
ase let me knoww if not.']
```

```
In [11]: # applying the stop words removal and typo correction
         correct_sentence_list = []
         for sent in sentence_list:
             correct_sentence_list.append(' '.join([spell.correction(w
         d) for wd in word_tokenize(sent.lower()) \
                                                if wd not in stopword_list+p
         unctuation_list]))
```

```
In [12]: correct_sentence_list
```

```
Out[12]: ['class', 'hope interesting', 'please let know']
```

```
In [13]: #unigram
         vectorizer = CountVectorizer(ngram_range=(1,1))
         X = vectorizer.fit_transform(correct_sentence_list)
         df = pd.DataFrame(X.toarray())
         df.columns = vectorizer.get_feature_names()
         df
```

Out[13]:

	class	hope	interesting	know	let	please
0	1	0	0	0	0	0
1	0	1	1	0	0	0
2	0	0	0	1	1	1

```
In [14]: # Tf-Idf transformation of unigram
         vectorizer = TfidfVectorizer(ngram_range=(1,1))
         X = vectorizer.fit_transform(correct_sentence_list)
         df = pd.DataFrame(X.toarray())
         df.columns = vectorizer.get_feature_names()
         df.round(2)
```

Out[14]:

	class	hope	interesting	know	let	please
0	1.0	0.00	0.00	0.00	0.00	0.00
1	0.0	0.71	0.71	0.00	0.00	0.00
2	0.0	0.00	0.00	0.58	0.58	0.58

Fig. 14.3 Code examples of creating unigram representation and TF-IDF transformation from a list of three example sentences

We will introduce the LSA in the following section and leave the LDA as exercise for readers.

The underlying assumption for LSA to work is the distributional hypothesis in linguistics, which assumes words and expressions that occur in the same contexts tend to have similar meaning (Harris, 1954). The transpose of the document-term matrix is a term-document matrix whose rows index the terms and columns index the documents. The term-document matrix is generally used in the discussions of LSA. After applying a TF-IDF transformation to the term-document matrix, each row of the matrix is a weighted frequency of the corresponding term appearing in different documents while each column is a weighted frequency of all the terms appearing in the corresponding document. For a term-document matrix X with d rows (documents) and t columns (terms), a matrix decomposition technique, known as Singular Value Decomposition (SVD) in linear algebra, can decompose it as the product of three matrices as follows:

$$X = U \Sigma V^T$$

Where U is a t by r matrix and V is a d by r matrix with r is the rank of X. Both U and V have orthonormal columns, e.g., $UU^T = VV^T = I$. Σ is a r by r diagonal matrix of singular values that are non-negative and ordered in decreasing magnitude by convention.

SVD provides a way to find a lower rank matrix to approximate the term-document matrix. If we consider only the part of Σ that contains the largest k singular values (e.g., k topics in the LSA context), we have a $\hat{X} = U_k \Sigma_k V_k^T$, where U_k is a t by k matrix, Σ_k is a k by k matrix, and V_k is a d by k matrix. The U_k matrix contains information regarding how the k topics are related to the terms and the V_k matrix contains information on how the k topics are related to the documents, as shown in Fig. 14.4.

LSA provides a way to extract topics that are linear combinations of the original terms and are orthogonal to each other, which helps to uncover information in large collections of text. It has also been shown that queries performed in the topic space outperform those in the original term space (Deerwester et al., 1990), which was the initial motivation of the introduction of LSA in the context of information retrieval. The space spanned by the topics is smaller than the one spanned by the terms, which means some degree of dimensional reduction has been achieved. Using LSA, document similarity and clustering can be performed with respect to any given selected subset of topics.

14.3 Automated Scoring

Automated scoring is one of the most popular applications of NLP in educational assessments. Broadly speaking, automated scoring is a specific type of task whose goal is to assign a score or a label to a text, typically scoring constructed responses

	Topic 1	*Topic 2*	...	*Topic k*
Term 1	…	…	...	...
Term 2	…	…	...	...
Term 3	…	…	...	...
...	…	…	...	...
Term t	…	…	...	...

	Topic 1	*Topic 2*	...	*Topic k*
Document 1	…	…	...	...
Document 2	…	…	...	...
Document 3	…	…	...	...
...	…	…	...	...
Document t	…	…	...	...

Fig. 14.4 The structure of U_k (upper panel) and V_k (lower panel)

such as student essays. An automated scoring engine usually has two major components. The first is a feature representation that turns the raw texts into numerical forms, and the second is to map the feature representation to a numerical score scale or categories via statistical modeling or machine learning methods. Some of the features can be directly computed from text, for example the word count, or the type/token ratio.[5] Computing other features may require external resources. For example, to find out how many common and rare English words are contained in an essay, an external database of word frequencies would be required. Still other features are often more context-specific.[6] For example, in order to compute how many nouns are there in an essay, a text typically needs to be processed with a part-of-speech tagger, which usually considers the context of each word during

[5]Number of different words by number of all words (the footnote to the left has 9 tokens but only 6 types).

[6]In the sense that analysis of each word may require considering neighboring words and possibly also going beyond the sentence boundaries (for example for computing inter-sentence cohesion of a text).

processing. An automated scoring engine may use different level of combinations of features extracted from the text, depending on the type of scoring task at hand.

The notion of automated scoring covers two related but distinct areas. On the one hand, Automated Essay Scoring is an area of research and applications that is specifically focused on evaluation of writing quality and the estimation of the students' proficiency in writing. On the other hand, Content Scoring, is an area of research and applications for assessment of the students' knowledge and opinions on a particular subject matter. The distinction is between assessing writing proficiency and assessing the correctness/completeness of a response to a question. In the following subsections, we will introduce some details of both strands of scoring. We are going to give two specific examples of modern scoring systems, but we note that there are many other approaches that are not mentioned here.

14.3.1 Automated Essay Scoring

Automatic essay scoring (AES) refers to the use of computer technology to evaluate and score students' essays. This area is also known as Automated Essay Grading (AEG) and Automated Writing Evaluation (AWE). The concept of automatic scoring of essays was introduced by Ellis Page in 1966 (Page, 1966). By 1973, the first successful AES system, Project Essay Grade (PEG) was implemented (Page & Petersen, 1995). In the 1990s, AES became a practical and economically viable approach, with the proliferation of electronic texts and ubiquity of computers for student writing. For surveys on the history and utility of AES, see Shermis and Burstein (2013) and Dikli (2006). Since then, AES has been widely used to score constructed responses in educational tests (Ke & Ng, 2019; Vitartas et al., 2016; Zupanc & Bosnic, 2016).

The typical advantages of AES over human scoring are low cost, fast turnaround, and consistency over time (Williamson et al., 1999). Additional advantages are the constant availability of scoring and reduction of the need to coordinate large teams of raters. Another important advantage of AES is the potential for real-time feedback to students (Williamson et al., 2012), where feedback may include the scores and also detailed comments and highlights of errors.

Development of modern AES systems is based on collecting empirical data (essays scored by human raters), and automatically (or semi-automatically) inferring from such data how essays should be scored. This typically involves a supervised machine learning task, where feature weights are tuned and parameters are estimated to producing a single score for an essay (Valenti et al., 2003). The role of NLP is to prepare texts and extract features, over which the machine learning can operate. There is a large variety of approaches, varying in what features are extracted and how the learning process is conceptualized. Essay scoring can be viewed as a regression problem (Attali & Burstein, 2006), as a classification task (Rudner and Liang, 2002), or as hierarchical classification (McNamara et al., 2015). Classification approaches develop models for each level on a nominal grading scale;

the text of a new response is then compared to the model of each grading level, and is assigned the score of the model to which it is most similar.

One particular implementation of this approach uses the LSA technique we introduced above (Landauer et al., 1998), which has been implemented in a system called Intelligent Essay Assessor (Landauer et al., 2003). In-LSA-based approaches, an NLP system first accumulates a large matrix of word-document co-occurrences, the matrix is then mathematically transformed. As a result, the system learns to associate a word with a vector of numbers that serves as its 'meaning representation'. To build a scoring model, human scored essays are collected. For each essay, the words are collected, converted to vectors and aggregated.[7] A new essay that needs to be scored is similarly converted to a set of vectors. Then, the task is to find to which model the new essay is most similar – by comparing the aggregated vector components (Kakkonen et al., 2005; Landauer et al., 2003).

A different, regression-based approach has been developed at the Educational Testing Service. e-Rater™ is a computer program that scores essays primarily on the basis of writing quality by means of feature values computed using natural language processing. A distinguishing aspect of e-Rater is that it uses a two-tier approach to aggregating components (Attali & Burstein, 2006). For the first tier, a large number of features is computed using NLP. The features of the first tier range from rather simple, such as average word length and median word frequency per essay, number of spelling errors, to more sophisticated, such as specific kinds of grammar and usage errors, etc. Grammar-error features include errors in subject-verb agreement, preposition errors, pronoun errors, article errors, sentence fragments, missing commas, wrong word forms, repetition of words, etc. For the second tier, some of the features are organized along the lines of trait scoring, which recognizes that essay quality has several dimensions, such as grammar, mechanics, style, organization, etc. (Attali et al., 2010; Quinlan et al., 2009; Ramineni & Williamson, 2013). Thus, some of the features are aggregated into macro-features.[8] e-Rater uses samples of human-scored essay data for building scoring models. First, features are computed. This is followed by aggregation of features into macro-features, based both on factor analysis and linguistic understanding of where different features belong. Then, the process uses multiple regression analysis to determine what weighting scheme of macro-features best predicts the human scores (Chen et al., 2017; Deane & Quinlan, 2010; Ramineni & Williamson, 2013).

The macro-features and micro-features have two applications. The macro-features are used directly for automated scoring. The individual micro-features are used to compute the macro-features, but they can also be used for formative

[7] A text document is typically represented as a vector which is a sum or an average of the vectors of its words..

[8] The specific methods of aggregation or combination may vary.' Some may involve weighted summation of feature values, while some may include complex transformations of feature values. Features that are combined into larger assemblies are called 'micro-features, and the larger assemblies are called macro-features. In e-Rater, a feature, or an assembly, whose values are used directly in a scoring formula is called 'macro-features'.

feedback to inform students about specific errors in grammar, usage, mechanics, style, and vocabulary, while macro-features may be useful for feedback on essay organization and development (Dikli & Bleyle, 2014; Stevenson, 2016).

The above process is typically applied for writing tasks where the goal is to assess the general writing ability of examinees. Note that topics for writing tasks can be relatively broad or narrow. For writing tasks that are focused on narrow topics, the scoring models may involve additional features that model the usage of topic-specific vocabulary, reflecting the specific content (Beigman Klebanov et al., 2016; Deane & Quinlan, 2010).

14.3.2 Automated Scoring of Short Constructed Responses

Assessment of student knowledge (especially in subject areas, such as biology or history) has been traditionally done via multiple-choice questions. An alternative approach is to let the students express their knowledge and opinion on their own, in their own words (Bennett et al., 1990), i.e. via student-constructed responses. Automated scoring of student responses is known as Automated Short Answer Grading (ASAG), Short Answer Scoring (SAS), content scoring or content-focused assessment. Short constructed response items are widely used for assessing knowledge and understanding (Shermis, 2015). There are two main approaches to the automated scoring of short responses (Horbach & Zesch, 2019).

An early approach for short answer scoring was rule based. The experts used testing rubrics and specified what kind of content (words and expressions) needed to be found in a response and how many points should be assigned for the expected content (Leacock & Chodorow, 2003; Sukkarieh & Blackmore, 2009). However, such an approach has certain disadvantages: preparation of models is cumbersome and time-consuming, experts need to anticipate a variety of potential expressions for the required content; thus it is costly, slow to develop, and prone to underestimating the variability in constructed responses.

The rule-based approach has then morphed into a more manageable 'instance' based approach (Okoye et al., 2013). Instead of supplying rules for scoring, the expert can provide one or more exemplar responses, and an automated system scores student responses by computing how similar the responses are to the gold-standard reference. The NLP system typically breaks the reference exemplar and the student responses into components (words, phrases, n-grams, or abstracted representations, such as vectors), and applies automated similarity measurement techniques.

A more modern approach uses similarity and machine learning. Instead of computing the similarity of a new response to the gold standard reference, a system

can be trained to measure similarity to multiple previously-scored responses and automatically derive the proper score.[9]

In order to build an effective machine-learning-based scoring system, a lot of human-scored responses are needed (usually at least a few hundred per prompt). The role of NLP is then to convert the responses into quantified features that can be fed to the machine learning approaches (Burrows et al., 2015; Madnani et al., 2017). In a more recent approach, neural networks and deep learning methods are used for the same task (see Horbach & Zesch, 2019, and Riordan et al., 2017, for refcent reviews).

14.4 Summary

In this chapter, we present a concise overview of NLP and introduce some typical applications in education. We hope this introduction can provide researchers with a psychometrics background a general picture of the application of NLP in educational contexts. As NLP itself is a mature scientific discipline and is undergoing rapid progress due to the revolutionary advances in deep-neural-network-based methods in recent years, we recommend readers consult the dedicated volumes (e.g., Jurafsky & Martin, 2019) for more in-depth knowledge.

Acknowledgement The authors thank Beata Beigman Klebanov, Aoife Cahill and Isaac Bejar for helpful comments on the manuscript, and Rick Meisner for copyediting.

Bibliography

Attali, Y., Bridgeman, B., & Trapani, C. (2010). Performance of a generic approach in automated essay scoring. *The Journal of Technology, Learning and Assessment, 10*(3).

Attali, Y., & Burstein, J. (2006). Automated essay scoring with e-rater® V. 2. *The Journal of Technology, Learning and Assessment, 4*(3).

Basili, R., & Magnini, B. (2012). Natural language processing in the web era. *Intelligenza Artificiale, 6*(2), 117–119.

Beigman Klebanov, B., Flor, M., & Gyawali, B. (2016). Topicality-based indices for essay scoring. In *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications (BEA11)*, pages 63–72.

Bennett, R. E., Ward, W. C., Rock, D. A., & LaHart, C. (1990). *Toward a framework for constructed-response items. Research report*. Educational Testing Service.

[9] A basic approach is to compare a new essay with the best-scoring previous essays, and to derive a score from the similarity to best essays. Another approach is to compare the new essay to previously scored essays from different score points, so a new essay gets a score from the group to which it is most similar. Previously-scored essays here are criterial data, typical scored by expert raters.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research, 3*(Jan), 993–1022.

Burrows, S., Gurevych, I., & Stein, B. (2015). The eras and trends of automatic short answer grading. *International Journal of Artificial Intelligence in Education, 25*(1), 60–117.

Chen, J., Zhang, M., & Bejar, I. I. (2017). An investigation of the e-rater® automated scoring Engine's grammar, usage, mechanics, and style microfeatures and their aggregation model. *ETS Research Report Series, 2017*(1), 1–14.

Chiticariu, L., Li, Y., & Reiss, F. (2013). Rule-based information extraction is dead! Long live rule-based information extraction systems!. In *Proceedings of the 2013 conference on Empirical Methods in Natural Language Processing*, pages 827–832.

Deane, P., & Quinlan, T. (2010). What automated analyses of corpora can tell us about students' writing skills. *Journal of Writing Research, 2*(2).

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science, 41*(6), 391–407.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). *BERT: Pre-training of deep bidirectional transformers for language understanding*. arXiv preprint arXiv:1810.04805.

Dikli, S. (2006). An overview of automated scoring of essays. *The Journal of Technology, Learning and Assessment, 5*(1).

Dikli, S., & Bleyle, S. (2014). Automated Essay Scoring feedback for second language writers: How does it compare to instructor feedback? *Assessing Writing, 22*, 1–17.

Feldman, R. (2013). Techniques and applications for sentiment analysis. *Communications of the ACM, 56*(4), 82–89.

Harris, Z. S. (1954). Distributional structure. *Word, 10:*(2–3), 146–162, https://doi.org/10.1080/00437956.1954.11659520

Horbach, A., & Zesch, T. (2019). The influence of variance in learner answers on automatic content scoring. *Frontiers in Education, 4*, 28.

Hutchins, J. (1995). Machine translation: A brief history, concise history of the language sciences: From the Sumerians to the cognitivists. Edited by EFK Koerner and RE Asher.

Jurafsky, D & Martin, J. H. (2008). *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Prentice Hall.

Jurafsky, D & Martin, J. H. (2019). Speech and Language Processing (3rd ed. draft). https://web.stanford.edu/~jurafsky/slp3/

Kakkonen, T., Myller, N., Timonen, J., & Sutinen, E. (2005, June). Automatic essay grading with probabilistic latent semantic analysis. In Proceedings of the second workshop on building educational applications using NLP (pp. 29–36).

Ke, Z., & Ng, V. (2019, August). Automated essay scoring: A survey of the state of the art. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence* (pp. 6300–6308). AAAI Press.

Kolomiyets, O., & Moens, M. F. (2011). A survey on question answering technology from an information retrieval perspective. *Information Sciences, 181*(24), 5412–5434.

Landauer, T. K., Foltz, P. W., & Laham, D. (1998). Introduction to latent semantic analysis. *Discourse Processes, 25*, 259–284.

Landauer, T. K., Laham, D., & Foltz, P. W. (2003). Automated scoring and annotation of essays with the Intelligent Essay Assessor. In M. D. Shermis & J. Burstein (Eds.), *Automated essay scoring: A cross-disciplinary perspective* (pp. 87–112). Erlbaum.

Leacock, C., & Chodorow, M. (2003). C-rater: Automated scoring of short-answer questions. *Computers and the Humanities, 37*(4), 389–405.

Levy, O., & Goldberg, Y. (2014). Neural word embedding as implicit matrix factorization. In *Proceedings of Advances in Neural Information Processing Systems, NeurIPS 2014*. https://proceedings.neurips.cc/paper/2014/file/feab05aa91085b7a8012516bc3533958-Paper.pdf

Li, H. (2017). Deep learning for natural language processing: Advantages and challenges. *National Science Review, 5*(1), 24–26. https://doi.org/10.1093/nsr/nwx110

Madnani, N., Loukina, A., & Cahill, A. (2017, September). A large scale quantitative exploration of modeling strategies for content scoring. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications* (pp. 457–467).

Manning, C. D. (2015). Computational linguistics and deep learning. *Computational Linguistics, 41*(4), 701–707.

Manning, C. D., Manning, C. D., & Schütze, H. (1999). *Foundations of statistical natural language processing*. MIT press.

McNamara, D. S., Crossley, S. A., Rosco, R. D., Allen, L. K., & Dai, J. (2015). A hierarchical classification approach to automated essay scoring. *Assessing Writing, 23*, 35–59.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pp. 3111–3119.

Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., & Joulin, A. (2018). Advances in pre-training distributed word representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. Miyazaki, Japan. https://www.aclweb.org/anthology/L18-1008

Mosteller, F., & Wallace, D. L. (1963). Inference in an authorship problem. *Journal of the American Statistical Association, 58*(302), 275–309.

Okoye, I., Bethard, S., & Sumner, T. (2013, June). CU: Computational assessment of short free text answers-a tool for evaluating students' understanding. In Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013), at the Second Joint Conference on Lexical and Computational Semantics (* SEM), pages. 603–607.

Page, E. B. (1966). The imminence of ... grading essays by computer. *Phi Delta Kappan, 47*(5), 238–243.

Page, E. B., & Petersen, N. S. (1995). The computer moves into essay grading: Updating the ancient test. *Phi Delta Kappan, 76*(7), 561.

Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval, 2*(1–2), 1–135.

Pennington, J., Socher, R., & Manning, C. D. (2014). GLoVe: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). *Deep contextualized word representations*. arXiv preprint arXiv:1802.05365.

Quinlan, T., Higgins, D., & Wolff, S. (2009). Evaluating the Construct-Coverage of the e-rater® Scoring Engine. Report RR-09-01, ETS Research Report Series. : Educational Testing Service.

Ramineni, C., & Williamson, D. M. (2013). Automated essay scoring: Psychometric guidelines and practices. *Assessing Writing, 18*(1), 25–39.

Riordan, B., Horbach, A., Cahill, A., Zesch, T., & Lee, C. (2017). Investigating neural architectures for short answer scoring. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications* (pp. 159–168).

Rudner, L. M., & Liang, T. (2002). Automated essay scoring using Bayes' Theorem. *The Journal of Technology, Learning and Assessment, 1*(2), 3–21. https://ejournals.bc.edu/index.php/jtla/article/view/1668

Shermis, M. D. (2015). Contrasting state-of-the-art in the machine scoring of short-form constructed responses. *Educational Assessment, 20*(1), 46–65.

Shermis, M. D., & Burstein, J. (Eds.). (2013). *Handbook of automated essay evaluation: Current applications and new directions*. Routledge.

Slocum, J. (1985). A survey of machine translation: Its history, current status and future prospects. *Computational Linguistics, 11*(1), 1–17. https://www.aclweb.org/anthology/J85-1001

Stevenson, M. (2016). A critical interpretative synthesis: The integration of automated writing evaluation into classroom writing instruction. *Computers and Composition, 42*, 1–16.

Sukkarieh, J. Z., & Blackmore, J. (2009). C-rater: Automatic content scoring for short constructed responses. In *Proceedings of the 22nd International Florida Artificial Intelligence Research Society Conference*, pages 290–295.

Tur, G., & De Mori, R. (2011). *Spoken language understanding: Systems for extracting semantic information from speech*. John Wiley & Sons.

Valenti, S., Neri, F., & Cucchiarelli, A. (2003). An overview of current research on automated essay grading. *Journal of Information Technology Education, 2*. https://doi.org/10.28945/331

Vitartas, P., Heath, J., Midford, S., Ong, K. L., Alahakoon, D., & Sullivan-Mort, G. (2016). Applications of automatic writing evaluation to guide the understanding of learning and teaching. In *Proceedings of 33rd International Conference of Innovation, Practice and Research in the Use of Educational Technologies in Tertiary Education*, pages. 592–601.

Wang, S. (2005). Corpus-based approaches and discourse analysis in relation to reduplication and repetition. *Journal pf Pragmatics, 37*, 505–540.

Wikipedia Contributors. (2019, June 3). Text mining. In Wikipedia, The Free Encyclopedia. Retrieved 18:10, June 26, 2019, from https://en.wikipedia.org/w/index.php?title=Text_mining&oldid=900109344

Williamson, D. M., Bejar, I. I., & Hone, A. S. (1999). 'Mental model' comparison of automated and human scoring. *Journal of Educational Measurement, 36*(2), 158–184.

Williamson, D. M., Xi, X., & Breyer, F. J. (2012). A framework for evaluation and use of automated scoring. *Educational Measurement: Issues and Practice, 31*(1), 2–13.

Zupanc, K., & Bosnic, Z. (2016). Advances in the field of automated essay evaluation. *Informatica, 39*(4), 383–395.