Bias by default? A means for *a priori* interface measurement

Joseph A. Cottam*
Pacific Northwest National Laboratory

Leslie M. Blaha†
Pacific Northwest National Laboratory

ABSTRACT

Systems have biases. Their interfaces naturally guide a user toward specific patterns of action. For example, modern word-processors and spreadsheets are both capable of handling word wrapping, checking spelling, and calculating formulas. You *could* write a paper in a spreadsheet or *could* do simple business modeling in a word-processor. However, their interfaces naturally communicate which function they are designed for. Visual analytic interfaces also have biases. We outline why simple Markov models are a plausible tool for investigating that bias, even prior to user interactions, and how they might be applied to understand *a priori* system biases. We also discuss some anticipated difficulties in such modeling and touch briefly on what some Markov model extensions might provide.

Index Terms: H.5.2 [Information Systems]: Information Interfaces and Presentation—User Interfaces; H.1.2 [Information Systems]: Models and Principles—User/Machine Systems

1 INTRODUCTION

Does all data in an application have an equal chance of being seen? The answer to this question is likely "no", and that is not necessarily a bad thing. We deliberately influence what is visible and what is not based on many measures. In fact, we rely on such imbalances as part of the data exploration process to keep the information content tractable for human memory and reasoning [14]. Any time something "just pops out" or is "obvious" in a display, there is an element of bias at play. However, does the interface naturally bias in the direction of tasks it was deigned to support? How much of that bias is inherent in the interface, and how much is the result of the ways the interface interacts with a specific dataset? How much is the result of the user crafting the interface for personal needs and interests? This paper proposes Markov modeling as an approach to begin teasing apart the sources of bias in visual analytic systems.

Friedman and colleagues defined bias in computers systems as a slant which produces systematic and unfair discrimination against certain individuals or groups, particularly when that discrimination is paired with unfair outcomes [9, 10]. They defined three types of biases: pre-existing, technical, and emergent. Although we disagree that bias only produces unfair outcomes, we find these classes useful for thinking about bias that can estimated about the system with and without user interactions. Pre-existing bias reflects how a system embodies cultural norms, practices, and attitudes that exist in the environment in which the system was developed, programmed, or deployed. Pre-existing biases in visual analytics might reflect the culture of the company or research group that developed the system. They could be as simple as the default interface elements, like default color schemes or variable placements on axes.

Other system biases are technical biases. These arise from technical constraints or considerations in the design process, such as choice of hardware or peripherals, which shape the capabilities of the system. Technical bias in visual analytic systems can influence the initial layout, the available algorithms, or the options for interaction techniques. Interaction options have implications for the amount of information that needs to be available on screen. For example, hover and roll-over functions may not be enabled without a mouse or touchpad. Without a hover option, tool tips may not be possible, so information that might have been available on demand may need to be readily available in other ways or on the screen at all times. Or the burden can be placed on the user to query for the information; however, if the user is inexperienced with the system or poor at formulating queries, then some information may not be queried and so may not be seen. Another form of technical bias can be seen in the specific algorithms provided in a tool. They are often chosen based on expected performance on reference hardware for expected datasets. As hardware advances, previously intractable algorithms can be implemented, and as new datasets are approached with a tool, different algorithms may be preferred.

A third class of system biases are emergent biases that result from the interactions of users with the system. These are very much of interest to visual analytic systems which are meant to facilitate extensive interactions for data exploration [3]. However, we suspect that emergent biases can only be measured from user interactions with the system. This is because each user has unique biases from attitudes, experience, and task goals that will shape the emergent biases. Whether the goal of measurement is online or *post hoc* bias assessment, it is hard to predict emergent biases in the absence of specific user characteristics and interaction behavior data.

Thus, the goal of this paper is to propose a framework by which we can measure the biases of an interface from the design of the system, including choices of visualizations and interactions. This may include elements of both technical and pre-existing biases, which do not require the collection of user interaction data for assessment. Of particular interest at present is predicting if the system design will steer users into system states where information is systematically unavailable or hard to recover, which will bias their exploratory reasoning and inference processes. Identifying the biases *a priori* helps (1) identify when and which biases are important, (2) compensate for biases when they hinder task performance, and (3) constructively employ biases when they help.

2 RELATIONSHIP TO ANALYTIC PROVENANCE

Modeling *a priori* system bias provides an important complement to analytic provenance modeling. The goal of provenance modeling is to leverage the sequence of user actions to characterize a user's analytic process [11, 16]. Xu and colleagues [16] argue that there are two important uses for analytic provenance: users can plan further analyses, and systems can suggest related but unexamined data. If captured and interpreted automatically, rather through intensive manual annotation, a mixed-initiative system could incorporate analytic provenance into intelligent recommendations, as illustrated by Endert, Fiaux, and North [6], Cook and colleagues [2]. Notably, Dabek and Caban [4] use captured actions to automatically build Markov-model-like automata that form the basis of their intelligent recommender system.

Additionally, when used *post hoc*, provenance enables analysts to study their own and others' processes. Toward this end, there have been efforts to develop visualizations for showing analytic provenance. GraphTrail [5] uses a graph visualization approach where the states of the analytic system are nodes, and the links

*e-mail: joseph.cottam@pnnl.gov
†e-mail: leslie.blaha@pnnl.gov

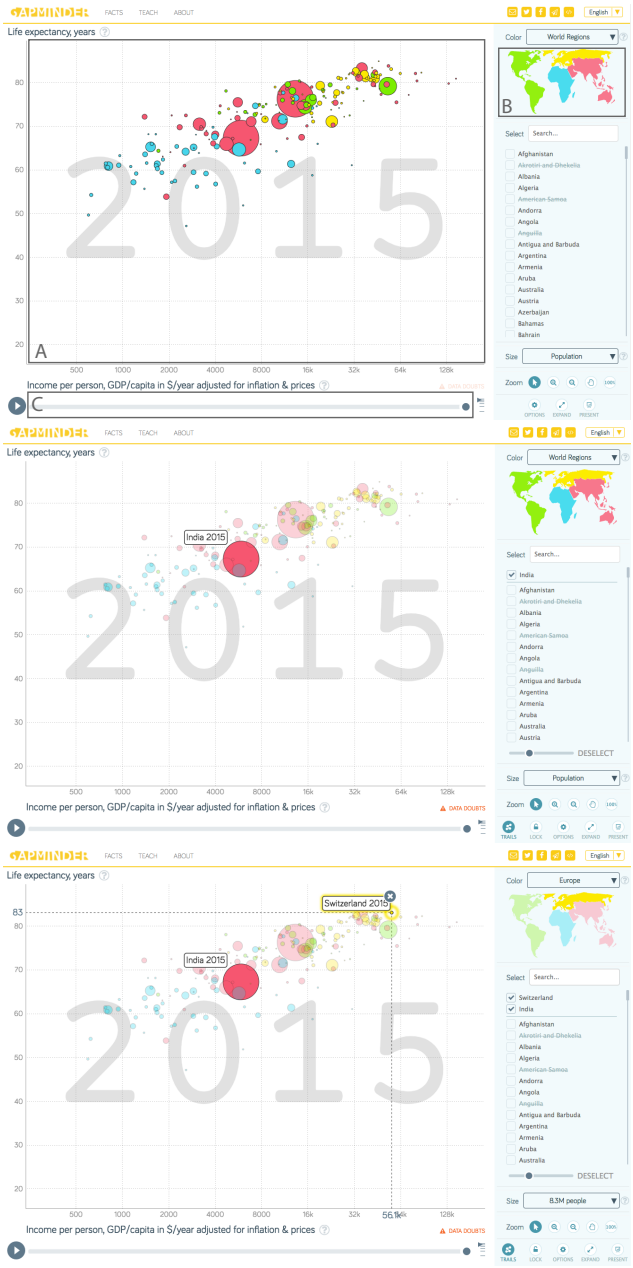

Figure 1: Series of images of a Gapminder "Bubbles" view in sequential states: (a) initial 2015 data, (b) select India, (c) hover Switzerland. Images from gapminder.org, CC-BY license.

illustrate the analysts transition path between the visualizations. Those links could be enriched by identifying the types of actions they represent in the analytic process, using the catalog of activity developed by Gotz and Zhou [11], for example.

From a system design perspective, analytic provenance analysis allows designers to inspect how design choices and interface elements were used throughout task completion. Our proposed Markov chain model for interface and exploration biases offers a predictive analysis for what might happen. This analysis can be conducted before the system is given to users; it can be engaged early and often in the design process. Importantly, our proposed interface and exploration bias computations are common across users, because they are about the system structure not the specific user interactions

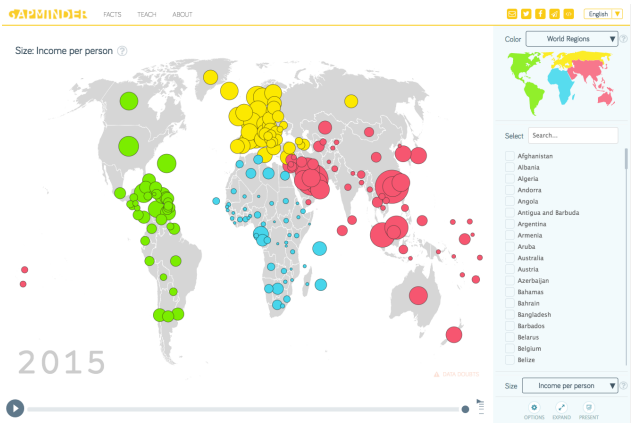

Figure 2: Gapminder map interface, dataset is the same as Fig. 1a.

or tasks. Thus, the emergent system biases introduced by the user interactions with the system may be teased apart from the other system biases by leveraging a combination of Markov-chain-based interface analyses and analytic provenance modeling.

Analytic provenance can then capture what a user *actually* does with a system, and the two can be compared. We propose that additional modeling the system absent user interactions targets measuring the potential biases in the system that would influence the ways a user *could* or *should* use the system. Technical or pre-existing biases may create some systems states that are not useful or would strongly sway the analytic process. While we can observe if or when analysts navigate into those states using analytic provenance, *a priori* modeling may help us to predict or prevent states unhelpful to the sensemaking process, or that might be compounded by user biases to create strong emergent biases.

3 Markov Models

We propose that Markov chains can be used to model user interfaces and reveal potential biases in those interfaces. That is, we can model interface changes as a probabilistic sequence through a system's state space. We focus on the visual states that can be observed, leaving aside state changes that are only based on hidden internal representation changes.

A general *Markov model* is a statistical process that can be represented as a sequence of states and transition probabilities between those states (i.e., a state machine). Formally, let S_i for $i = 1,\ldots,n$ be a set of n possible states, and we define $P(S_i|S_j) = p_{ji}$ as the transition probability from state S_j to state S_i. A sequence of states may be thought of formally as $\{S_i, S_j, S_k, S_i, \ldots\}$, where a repeated state, like S_i represents re-visiting a state. All Markov models adhere to the *Markov property*, which means transitions only depend on the current state (also called being "memoryless"). We represent this as the state of the system at time t being only a function of the state at time $t-1$, $P(S_t|\{S_1, S_2, \ldots, S_{t-1}\}) = P(S_t|S_{t-1})$.

The state machine model is the basis for other Markov processes. For example, a Markov chain is a path through a Markov model [13]. Hidden Markov models are Markov models that maximize the probability of observed chains when the underlying state space and probabilities are not known [1]. Markov models are "simple" in that they are amenable to many different kinds of analyses that yield useful information. Therefore, building a Markov model that faithfully reproduces system behaviors can lead to useful insights about expected behaviors under other circumstances.

The sequence of states in a Markov chain can represent a sequence of states the visual interface can go through. Those state changes maybe driven by direct user actions, streaming data updates,

or mixed-initiative analysis as it makes recommendations. The complete set of states in the Markov model is comprised of the union of all valid chains. This concept is illustrated in Figure 1. We note that a display changes can result from a change in either the content/data (e.g., the data is "played" through time in Gapminder) or a change in the layout or design parameters (e.g., the visualization is reconfigured in the right-side panel). To supply the transition probabilities, and thereby complete the Markov model, we assume that possible states of the interface are states in the Markov model and transition probabilities are derived from the screen presence of interface elements. A user session is a chain, drawn from the probability space defined by the model. Analyzing the Markov model state machine provides insight into possible and probable user session patterns.

We can gain insight about potential system biases, pre-existing and technical, by examining the structure of connections between and understanding the relative likelihoods of interface states. For example, some states may not be reachable without a specific sequence of user actions, making them less likely to occur. Other states may have likelihoods that change over time because of certain design or algorithm choices. Still others may be dependent on the default settings (the initial conditions) of the system. Modeling the user interface independent of actual user actions provides a basis for comparing interfaces to each other. Additionally, examining user interface actions in light of interface bias can tell you if observed biases came from the tool or from the operator. It allows us to distinguish the potential technical and pre-existing biases from the emergent biases in interactive visual analytic systems.

4 Interface Models

There are at least two conditions to interface modeling: with and without data loaded. With a dataset loaded, we propose to construct the Markov model with three key features: (1) each link is a possible action; (2) each node is an interface state that results from an action, and (3) links are weighted proportional to the target area on the screen. The above procedure captures the essential idea, but it probably needs to be tempered in some cases. Figure 3 illustrates some of the network shapes that result from applying this process by hand to parts of the Gapminder "Bubbles" interface. Linear dependencies are evident, showing that moving large distances in time incurs many step costs, biasing the user to make comparisons in near neighborhoods.

Applying the same procedure to the Gapminder "Map" interface (Figure 2) yields similar patterns BUT with different weights. For example, in the "Bubbles" interface it is possible to directly select the Switzerland bubble (Figure 1c). However, in the map, Switzerland is completely occluded by neighboring data. A data-dependent analysis of the interface would directly reveal this bias against such data points by examining the weights derived from screen space. Similarly, data-dependent analysis could reveal if the bias *toward* particular data points is proportional to bias in the dataset.

We have only done a partial analysis of the Gapminder interfaces, but we expect similar patterns to be components of full-application analysis. Just observing structural patterns, these patterns can illustrate potential biases. For example, isolated groupings show areas that may be difficult to move between, a bias for staying with the current representation. Moving to more algorithmic analysis, it would be possible identify unreachable and difficult-to-access data.

Modeling with target-area proportional weights is at least partially justified by the Shannon entropy interpretation of Fitts' Law [8]. In brief, if more (or more unlikely) interface actions are required to reach a state, that state is less likely to be encountered by chance. A sequence of user actions can be viewed as a string that encodes the address of an interface state. In terms of information, if bits of information must be supplied to "address" a state, the likelihood of an error increases. If there are more redundant paths, it is analogous to address encoding redundancy, and the state is more likely.

The Markov chain conceptualization for data-dependent biases derives the transition probabilities, p_{ji}, from this weighting schema. We are capturing biases Where the transition probabilities shape Markov chains to end up in particular part of the state space or makes certain transitions more likely than others. With data in the system, we are measuring some of the technical biases. The data representations reflect the results of the underlying encoding/embedding schemes and choice of machine learning or analytic algorithms. These technical choices can bias the data available in the system. Pre-existing biases may come into play if the system is applied to data types for which it was not designed (e.g., applying numerical techniques to ineffectively encoded text data), because the norms and practices will not properly apply. But predominantly, data-dependent Markov chains capture technical system biases.

This preliminary analysis makes it evident that the basic procedure naïvely applied yield combinatorial explosion of states. For example, sequential data selection is done when picking specific countries in the Gapminder "Bubbles" chart. A full model is a lattice of all possible combinations of selections (A, B, C, A&B, A&B&C, A&C, B&C, etc.). For all but trivial examples, this is likely to be computationally intractable. Tempering full data dependence is probably necessary, and the focus of the next section. In truth, a mixture of data-dependent and data-independent modeling is likely to yield the best *tractable* models. Some of the simplifications used in Dabek and Caban [4] used to reduce the impact of redundant combinations may also have analogous simplifications for this *a priori* modeling.

Data Independent Modeling

Interesting patterns in the interface may be revealed by ignoring details of the data presentation. In the data-independent scenario, the resulting model is simplified but necessarily more abstract. It is constructed in the same way as the data-dependent bias case but with two simplifications. First, all interactions that directly involve the data are collapsed into a single link by type. For example, instead of a selection-related link for each data point, there is a single data-selection link. This necessarily implies that data-related states are also compressed together. The general transformation is shown in the left column of Figure 3. Second, because we are no longer considering the data representation, we can no longer use screen-space to weight the links. Instead, we propose to make all links that leave a node equally likely. This is termed a *regular* Markov chain, with the transition probability matrix $P = \left[\frac{1}{n}\right]$. This initial assumption provides a baseline against which we can study a system.

Data-independent Markov chains have transition probabilities that are regular or are shaped by the initial conditions of the system. If the transition probabilities are dependent on initial conditions, we are capturing a pre-existing bias in the system. That is, the assumptions made by the designer as to default settings produced a bias toward data availability that changed when those default settings were adjusted to some alternative initial configuration. Additional pre-existing biases are captured in the overall design elements in the display or choices of representation implemented, because all reflect some methodological attitude or cultural norm for that system. Technical biases can also be revealed if the data-independent display incorporates structures output from some internal algorithm, or the structure reflects technology choices on which the system is implemented. But we argue that data-independent Markov chains serve to capture pre-existing system biases.

Modeling an interface with a specific dataset represented it is likely to be more directly actionable than the data-independent model. However, the models are likely to be large relative to the data-independent case because many common interface patterns are combinatoric in the elements of the dataset. Working with the data-independent model has the effect of reducing the size the model significantly, but it makes the results more abstract and thus more difficult to interpret.

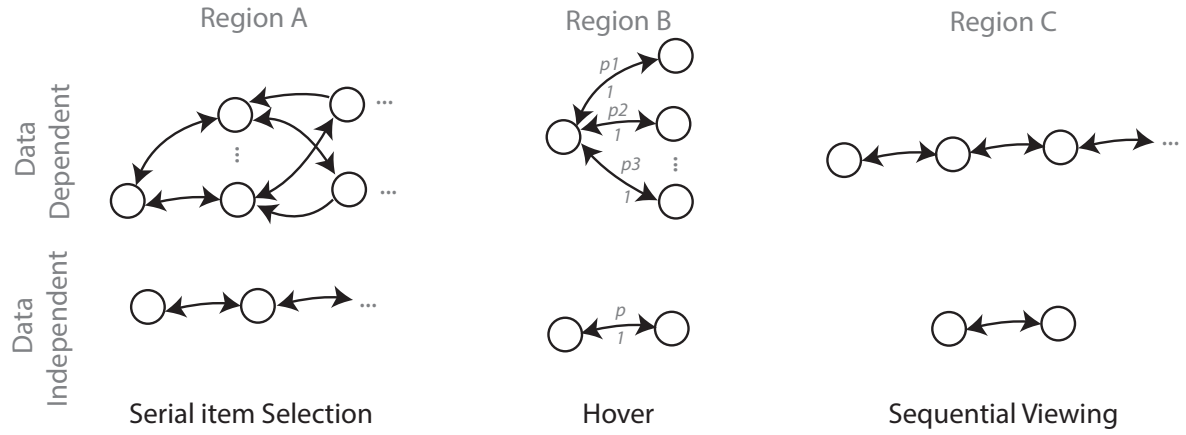

Figure 3: Markov model structures from Gapminder "Bubbles" regions noted in Fig. 1a. The difference between the data-dependent and -independent cases is evident in the difference of complexity between the rows.

5 Discussion

Classic Markov modeling is a "memory free" technique. It only takes the current state into consideration when making a transition. However, data exploration necessarily includes human memory [14]. Modeling multi-step memory with static Markov models is cumbersome at best (and practically impossible in combinatoric cases). However, compressing combinatoric cases into abstract chains (as discussed earlier) can be seen as a simple memory model. A similar compression technique might be used to model a simple form of memory. An alternative to combinatoric compression of states would be to use a model that includes memory in a structured way. Dynamic Markov, Push-down automata, and RAM-based automata (with limited RAM) are also be viable options. Each has an finite state space and a well-developed field of analysis.

Our proposed weighting scheme is simple, and may not be sufficient to illuminate some bias patterns. There are some interesting challenges. For example, in the data-dependent construction, the size-based weighting is derived from Fitts' Law. However, Fitts' Law not account for convention or attention. Therefore, some interface elements may be relatively large by convention but the probability that they will be interacted with is not proportional to their size. For example, menu bars have a size and position dictated by the interface guidelines of the platform, and that may be significantly larger than the representation of a single data point. Capturing such differences in the interaction probabilities requires reaching beyond Fitts' Law for transition probabilities.

In the data-dependent Markov modeling, *only* the screen real-estate is used to model direct data interactions. Logical extensions include using visual similarity (along many retinal dimensions) to up- or down-weight items. This could be extended further with a dynamic Markov model, so weights change based on what has been visited before. Proper dynamic weighting requires knowledge of the task as well as the visual representation. It makes sense to up-weight similar things when the retinal variables correspond to the desired task but to (possibly) down-weight similar items when the retinal variable does not have bearing on the task. Also, exploration versus verification probably have different interaction patterns. Such modeling may be achieved using a Markov Decision Process. In addition to a transition probability, the model is extended with a payoff matrix and a "discount" factor. Payoffs are provided when a specific transition is taken. The discount factor determines how important immediate versus future expected payoffs are weighted. Decisions are still based on the information observable in the current state, but the probability of a transition is made a factor of the base probability, the payoff, the expected future payoff, and the discount factor. Payoff and discount factors can be adjusted to model different goal-directed behaviors. Similar dynamic re-weighting is done in Dabek and Caban [4], captured in their "ideology" factors.

Analytic provenance models suggest another approach to Markov modeling. In particular, if a provenance tracking system records information about the state of the interface, we could use a hidden Markov model to derive the Markov chain of the original interface state space [7]. This might be helpful in cases where we have incomplete information about the structure or state space of an interface. This inference process could leverage existing graph modeling systems for analytic provenance, as in GraphTrail [5], to interpret the hidden model states. This approach bears some similarity to Jankun-Kelly's [12] P-set Model of visualization exploration. He defines two key concepts. A P-set is a set of parameters that define a visualization system, and visualization transformation is an operation on the P-set that creates a particular visualization view. Each set of parameter values (P-set) defines a state space with weighted connections (transformations) between the states. The difference between our Markov chain approach is that our links between the states quantify the probability of moving between states, rather than defining the parameter transformations themselves. An interesting direction for future work is to relate the transformations to transition probabilities between parameter states to capture emergent bias.

6 Conclusion

We note that methods for measuring information content in a visual analytic system remain an open challenge for the field [15]. Such measures are important for overall evaluation of systems, particularly for calibrating our expectations for how much information user's may be able to extract from a system. We propose that measurement of information availability and the interface biases that may shape that information availability should be modeled in systems before they are put into human-in-the-loop evaluations. Markov models, as proposed herein, provide a promising direction for conceptualizing the state space of a visual analytic system and understanding system-level biases through the transition probabilities over the state space.

Acknowledgments

This research was sponsored by the Analysis in Motion Initiative at the Pacific Northwest National Laboratory. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government.

REFERENCES

[1] D. H. Ballard. *An Introduction to National Computation*. MIT Press, Cambridge, MA, 1997.

[2] K. Cook, N. Cramer, D. Israel, M. Wolverton, J. Bruce, R. Burtner, and A. Endert. Mixed-initiative visual analytics using task-driven recommendations. In *Visual Analytics Science and Technology (VAST), 2015 IEEE Conference on*, pp. 9–16. IEEE, 2015.

[3] K. A. Cook and J. J. Thomas. Illuminating the path: The research and development agenda for visual analytics. 2005.

[4] F. Dabek and J. J. Caban. A grammar-based approach for modeling user interactions and generating suggestions during the data exploration process. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):41–50, Jan. 2017. doi: 10.1109/TVCG.2016.2598471

[5] C. Dunne, N. Henry Riche, B. Lee, R. Metoyer, and G. Robertson. Graphtrail: Analyzing large multivariate, heterogeneous networks while supporting exploration history. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1663–1672. ACM, 2012.

[6] A. Endert, P. Fiaux, and C. North. Semantic interaction for sensemaking: inferring analytical reasoning for model steering. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2879–2888, 2012.

[7] A. Endert, W. Ribarsky, C. Turkay, B. Wong, I. Nabney, I. D. Blanco, and F. Rossi. The state of the art in integrating machine learning into visual analytics. *Computer Graphics Forum*, 2017. doi: 10.1111/cgf.13092

[8] P. M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 47(6):381–391, 1954.

[9] B. Friedman. Value-sensitive design. *Interactions*, 3(6):16–23, 1996.

[10] B. Friedman and H. Nissenbaum. Bias in computer systems. *ACM Transactions on Information Systems (TOIS)*, 14(3):330–347, 1996.

[11] D. Gotz and M. X. Zhou. Characterizing users' visual analytic activity for insight provenance. *Information Visualization*, 8(1):42–55, 2009.

[12] T. Jankun-Kelly. Using visualization process graphs to improve visualization exploration. In *International Provenance and Annotation Workshop*, pp. 78–91. Springer, Berlin, 2008.

[13] J. G. Kemeny and J. L. Snell. *Finite Markov Chains*, vol. 356. van Nostrand, Princeton, NJ, 1960.

[14] R. E. Patterson, L. M. Blaha, G. G. Grinstein, K. K. Liggett, D. E. Kaveney, K. C. Sheldon, P. R. Havig, and J. A. Moore. A human cognition framework for information visualization. *Computers & Graphics*, 42:42–58, 2014.

[15] M. O. Ward, G. Grinstein, and D. Keim. *Interactive Data Visualization: Foundations, Techniques, and Applications*. CRC Press, Natick, MA, 2010.

[16] K. Xu, S. Attfield, T. Jankun-Kelly, A. Wheat, P. H. Nguyen, and N. Selvaraj. Analytic provenance for sensemaking: A research agenda. *IEEE Computer Graphics and Applications*, 35(3):56–64, 2015.