Good, but Not *That* Good: An Honestly-Noisy Visualization of Low-Fidelity Data Streams

Alvin Tan
UC Berkeley
Berkeley, CA, USA
alverino@berkeley.edu

Prabal Dutta
UC Berkeley
Berkeley, CA, USA
prabal@berkeley.edu

ABSTRACT

Long-term, ubiquitous sensing on wireless, power-limited devices requires aggressive data-reduction at the source to meet stringent networking and power constraints. However, the naive approach of error-triggered data updates obfuscates data and system information that is useful for downstream tasks. For example, understanding *error* and *stability* of outdoor temperature data is useful for those who are deciding what to wear in the morning. We constructed a hypothetical student-led deployment of low-fidelity temperature sensors across a university campus; designed a "noisy sensor" conceptual model to visually communicate the error and stability of the data; and compared our design against the naive baseline of displaying raw data values and a classic data visualization alternative of including a historical average. We then conducted an online survey with 150 participants and found that both the baseline and the classic alternative caused users to over-estimate accuracy of the data and stability of the underlying real-world temperature. Our noisy sensor design corrected these errors, but caused users to report false trends in the data. This study identifies need for continued work in developing task-based visualizations for low-fidelity data streams and in designing sensing systems that support them.

CCS CONCEPTS

• **Human-centered computing → Empirical studies in visualization**.

KEYWORDS

Ubiquitous sensing, data visualization, task-based design

ACM Reference Format:
Alvin Tan and Prabal Dutta. 2025. Good, but Not *That* Good: An Honestly-Noisy Visualization of Low-Fidelity Data Streams. In *The 3rd International Workshop on Human-Centered Sensing, Modeling, and Intelligent Systems (HumanSys '25), May 6–9, 2025, Irvine, CA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3722570.3726882

1 INTRODUCTION

Perpetual, long-term sensing systems built on low-power battery-agnostic devices [4, 13] and the growing presence of ubiquitous networks [1–3, 6, 11, 23] are creating a class of sensor systems that generate *live but sparse data streams.* These low-fidelity data streams have historically been collected for aggregate analysis or used in-house by developers who can properly interpret the data, but the incorporation of such sensing systems into smart homes and other consumer-level devices will require these low-fidelity data streams to be understandable by the general public. We claim this is nontrivial, and we spend the rest of this paper showing why.

To explore this idea, we constructed a hypothetical sensing scenario and conducted an online survey with 150 participants. We selected fine-grain temperature sensing as our application, due to its prevalence in the literature [8, 10, 12], familiarity to the general public, and expressed need among students and staff. We took real temperature data from LBNL weather stations [24], processed them, and presented them as data from small sensors scattered around campus. We assumed sensors used an *error-triggered upload strategy*, where the sensor sends an update if and only if the current temperature deviates more than a predefined "error threshold" away from the previously-sent value. This strategy is simple to implement, requires very little memory and computation, introduces negligible latency, and provides hard data-accuracy guarantees, making it very useful for low-power wireless sensor systems.

A brief task analysis highlighted the need for users to understand uncertainty around *data error* and *data stability*, so we designed our treatments to provide information about these two features.

We used three different treatments in our survey: `Naive`, `History`, and `Volatility`. `Naive` simply displays the collected data as-is, resulting in flat lines and abrupt steps when an update is received. `History` includes a 10-day average of the collected data to help users understand *historic* data stability. `Volatility` mimics a "noisy sensor," adding Gaussian noise proportional to the data's root mean square error, such that the subsequent noise floor obfuscates *current* data stability. We created `Volatility` through a task-based conceptual model design, which we describe in Section 3. Examples for each of these treatments are shown in Figure 1.

We presented these treated time-series data to students and staff on campus, and asked them to estimate the current temperature, future temperature, and error ranges for their responses. We found that `Naive` and `History` resulted in systematic under-estimation of data error, and that `Naive` obfuscated system shortcomings, resulting in unfounded confidence and satisfaction in the data at high error thresholds. `Volatility` successfully corrected these misinterpretations, but caused users to see unsubstantiated trends in the data. Thus, more work needs to be done to explore mechanisms for changing heuristics used in interpreting low-fidelity data streams.

The survey figures and survey responses used in this paper can be found at https://osf.io/r9e8u/.

HumanSys '25, May 6–9, 2025, Irvine, CA, USA

https://doi.org/10.1145/3722570.3726882

2 RELATED WORK

Our work is related to data visualization, stable dataset visualizations, and compression for downstream data pipelines.

Data Visualizations [5, 7, 9, 20] explore how data visualization styles distort viewer understanding of statistical properties of data. We draw on human-computer interaction (HCI) literature to design our task-based treatment.

Stable Dataset Visualizations [16, 18, 21, 22] highlight human-meaningful features of large datasets (e.g., average, min/max, trends) through coarse-grain data visualizations. Instead of retrospectively processing all the data, our work preemptively reduces data while preserving useful features.

Data Compression for Downstream Pipelines [17, 19, 25] optimize data compression algorithms for downstream machine learning or algorithmic (e.g., data forecasting) pipelines. Instead of optimizing for numerical analyses, our work optimizes for human consumption and understanding.

3 TASK-BASED DESIGN

We followed the HCI methodology [14, 15] of conducting a task analysis and generating a simple, task-based conceptual model to inform our subsequent data visualization.

Task analysis. Users use temperature data in many tasks:

- Deciding whether or not to bring an extra sweater,
- Choosing between going for a walk or staying inside during lunch break, and
- Seeing if that one room in that one building is going to be sweltering hot or freezing cold today.

These tasks require some sense of what the current temperature is, what the future temperature will be, and whether or not the presented data is "good" (i.e., expressive, accurate, fresh, etc.) enough to make a reasonable decision.

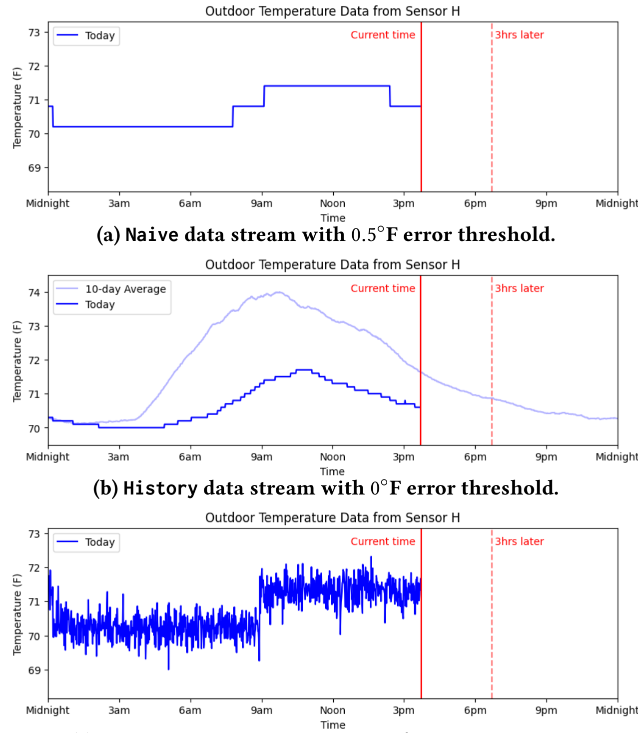
Task-oriented conceptual model. For a useful representation of the system, we want to make it clear that the temperature values are live, there is some amount of error in the visualized data, and the real-world temperature may be changing even if the change does not appear in the presented data. (Note that these statements are agnostic to the source of data errors.) We decided that a "noisy sensor" concept captures all of these – the sensor uploads noisy data all the time, so the data is live but there is uncertainty around the real-world value and small trends in the data may be hidden.

Data visualization. To emulate a noisy sensor, we considered either adding zero-mean uniform noise bounded by the error threshold or adding zero-mean Gaussian noise with the same standard deviation as the RMSE of the original data. We selected the Gaussian noise option because it looked more organic and because the uniform noise option overstated the expected error in the data.

4 USER SURVEY

We constructed a Qualtrics online survey and disseminated it through the Xlab platform[1] to UC Berkeley students and staff. The survey took less than 15 minutes to complete, and participants were compensated $5 for their time.

[1]https://xlab.berkeley.edu/



(a) `Naive` data stream with 0.5°F error threshold.

(b) `History` data stream with 0°F error threshold.

(c) `Volatility` data stream with 1.0°F error threshold.

Figure 1: Example figures presented in our user survey. Participants are asked to estimate current temperature, future temperature in three hours, and error ranges.

The premise of the survey was that some students planned to deploy temperature sensors on campus and wanted to see how good these sensors must be to be useful.

In the survey, participants were presented with time series temperature data from nine different sensors (examples in Figure 1) and asked to estimate the current temperature, the future temperature in three hours, and error ranges for how close their estimate is to real-life. We also asked how *confident* participants are that their answers "are good descriptions of real-world circumstances" and how *satisfied* the participants would be "to have this quality of data to use."

We end the survey with demographic questions about age, major, familiarity with sensors, and perceived difficulty of collecting temperature data across campus.

5 RESULTS AND ANALYSIS

We presented the survey to 177 participants. Of these, 21 did not answer any questions and 5 answered less than half of the questions. Additional data cleaning removed 1 participant and 13 individual responses (comprising 1.6% of total responses) due to obvious (e.g., off-by-10) typos. Thus, we conducted our analysis using 150 participants and generalized their responses to the broader population of "users" who do not have specialized systems knowledge. The distribution of responses used in the analysis are shown in Table 1. We then answered the full set of 108 survey questions

Error Threshold (°F)	Treatment		
	`Naive`	`History`	`Volatility`
0.0	99	101	138
0.5	113	119	100
1.0	118	98	110
1.5	109	117	115

Table 1: Number of responses for each combination of treatment and error threshold. There is some variance due to question randomization in the survey.

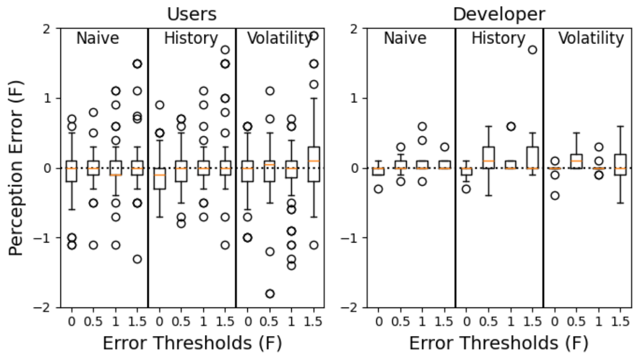


Figure 2: Perception error for users and developer, sign-normalized to compression error (i.e., positive values bias towards ground truth; negative values bias away). Users are much less accurate at reporting temperature values than a developer is. Neither user nor developer can consistently infer ground truth from low-fidelity data, since perception error is zero-median.

ourselves to use as "developer" responses. We assumed developers knew the mechanics of the treatments, but did not know the error thresholds *a priori* and must infer them from the data.

We found that (1) users interpret data *loosely*, *literally*, and *heuristically*, when compared to a developer, (2) `Volatility` improves both user and developer understanding of data error, compared to `Naive` and `History`, and (3) `Naive` obfuscates system shortcomings from the users, resulting in unwarranted confidence and satisfaction in the data.

5.1 Users interpret data *loosely, literally,* and *heuristically*

5.1.1 Loosely. User-reported values contained significant amounts of error. We decomposed error into two parts, such that $error_{total} = error_{comp} + error_{percep}$, where $error_{total}$ is the difference between the user-reported value and ground truth temperature, $error_{comp}$ is the difference between the value visualized in each figure (i.e., the compressed/uploaded data value) and ground truth temperature, and $error_{percep}$ is the difference between the user-reported value and the value visualized in each figure (i.e., perception error).

User and developer perception errors are shown in Figure 2. We see that user responses generally have larger interquartile ranges and more outliers than developer responses. This results in larger $error_{total}$ from ground truth, and may need to be accounted for when considering appropriate application- and systems-level accuracy values.

5.1.2 Literally. Users seemed to interpret the visualized data as literal ground truth, despite being explicitly prompted to infer real-world temperature values from the data. Specifically, users reported temperature values from the figure directly (i.e., minimizing $error_{percep}$ rather than $error_{total}$), inferred high temperature- and data-stability given visibly-flat plots, and systematically underestimated data error for large error thresholds.

Directly reading temperature values. Theoretically, a viewer with strong understanding of the system could infer $error_{comp}$ from the data and correct for it to minimize $error_{total}$. If they could do this, then we would expect their perception error to be biased towards ground truth values. Therefore, in Figure 2, we sign-normalized the perception errors such that positive values indicate bias towards ground truth while negative values indicate bias away from ground truth.

We see that although a developer is able to bias more towards ground truth than away, both users and developers are relatively zero-median and distribution, and neither are able to infer compression error consistently enough to achieve better-than-compression-error accuracy.

Inferring high temperature and data stability. In Table 2, we list error range estimates from the users and developer. We categorized range estimates into three types: positive ranges, indicating an estimate of non-zero error; zero range, indicating high confidence that the response is exactly ground truth; and negative ranges or a blank response, indicating inability to estimate any error for the data. Since percentages sum to 100% for each half-column, we omit the percentage of positive ranges and report their average values instead.

For `Naive` and `History`, the percentage of zero-range estimates increases significantly at high error thresholds. We hypothesize this is because large error thresholds result in fewer data updates and long, flat lines in the timeseries data, which users interpret to mean the temperature is particularly stable during that period of time, resulting in a predicted error range of 0. The developer easily avoids this mistake because they know that an error threshold exists.

Systematically underestimating data error. In Figure 3, we plot the distribution of the difference between the estimated (positive) error range and the actual error ($error_{total}$) for each response. Positive values indicate the range estimate was larger than the actual error, and negative values indicate the range estimate was smaller than the actual error.

For `Naive` and `History`, users systematically underestimate error as error threshold increases. This may be because users only account for perception error ($error_{percep}$) and do not consider $error_{comp}$ when estimating their error ranges. The developer, on the other hand, is able to infer the error threshold from step-like features in the data, and provides conservative range estimates for `Naive` and `History`.

5.1.3 Heuristically. Users are liable to provide estimates for error range and predict future trends in temperature that are not based in the data itself.

Error range. In Table 2, user estimates of positive error ranges for `Naive` and `History` are independent of the error threshold, staying around 0.35°F or 0.40°F on average. This is similar to the

	Treatment	`Naive`				`History`				`Volatility`			
	Error Threshold	0.0	0.5	1.0	1.5	0.0	0.5	1.0	1.5	0.0	0.5	1.0	1.5
Users	Mean of >0 ranges (°F)	0.36	0.35	0.36	0.35	0.33	0.42	0.41	0.38	**0.43**	**0.50**	**0.62**	**0.82**
	Amount of =0 ranges	7%	5%	14%	11%	3%	8%	8%	14%	**4%**	**2%**	**6%**	**3%**
	Amount not attempted	22%	25%	24%	21%	26%	19%	19%	20%	25%	19%	23%	35%
Developer	Mean of >0 ranges (°F)	0.34	0.47	0.83	1.18	0.30	0.50	0.80	1.04	0.31	0.41	0.69	0.98
	Amount of =0 ranges	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
	Amount not attempted	0%	22%	56%	56%	0%	0%	33%	44%	0%	0%	0%	0%

Table 2: Error ranges reported for current temperature estimates. Mean values are provided for positive ranges, since half-columns sum to 100%. Users interpret data literally and heuristically, inferring constant error ranges and high data stability for `Naive` and `History` as error threshold increases. `Volatility` corrects these misconceptions.

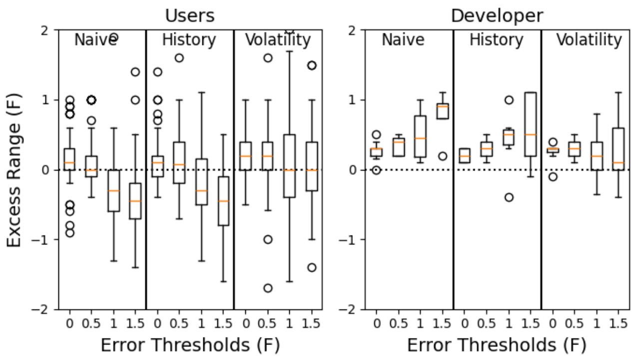


Figure 3: Difference between estimated error range and actual errors for users and developer. Positive values indicate error over-estimation; negative values indicate under-estimation. Users systematically under-estimate error for `Naive` and `History`, while developer systematically over-estimate. `Volatility` improves error estimates for both users and developers.

developer's estimate of perception error (i.e., error range estimates when error threshold is zero), which range from 0.30°F to 0.35°F.

Trends in temperature. For simplicity, we only consider the direction (and not the magnitude) of predicted change in temperature, and compare it to the ground truth change in temperature. We report the percentage of correct predictions, no predictions (i.e., current temperature and future temperature estimates were equal), and incorrect predictions in Table 3. `History` significantly improved user and developer ability to predict future change in temperature because it provided information about diurnal patterns in the data.

Interestingly, users predicted nonzero trends in temperature more often for `Volatility` than for `Naive`, despite `Volatility` providing no more information about temperature trends. Thus, the idea that "the real-world temperature may be changing even if the change does not appear in the presented data" may have been successfully conveyed through the `Volatility` visualization.

Even more surprising is that the additional predictions users make for `Volatility` are better than random. Compared to both the users' responses for `Naive` and the developer's responses for `Volatility`, the users' (unfounded) responses to `Volatility` in Table 3 are more right than wrong. Thus, users are likely using a heuristic (e.g., that temperatures increase in the morning and decrease in the evening) instead of interpreting the data directly.

5.2 `Volatility` improves both user and developer understanding of data error

5.2.1 For users. In Table 2, we see that `Volatility` helps users avoid the error of predicting zero error range, as discussed in Section 5.1.2, and helps users estimate error ranges that increase with error threshold.

In Figure 3, we see that users no longer systematically underestimate data error for high error thresholds, but provide error range estimates that are generally aligned with actual data error (i.e., zero-median), albeit with a notable amount of variance. Modulation of how much Gaussian noise is added may help shift the distribution of these error estimates in cases where e.g., conservative error estimates are preferred.

These improvements, with the insight from Section 5.1.3, give us cautious optimism that `Volatility` and the task-based design approach could be useful for providing users with proper understanding of sensing systems as a whole.

5.2.2 For developer. Developers are able to estimate the error threshold based on step-like features of the time series data, and thus estimate error ranges. However, the error threshold is an upper bound on the error, resulting in the overly conservative error estimates seen in Figure 3, and there may not be any steps in the time series data if the error threshold is particularly large, in which case the developer cannot give a range estimate, as seen in Table 2.

`Volatility` solves both of these issues. Since `Volatility` provides information on the variance of the data, which is a tighter bound than error threshold, we see in Figure 3 that estimated error ranges are more accurate (i.e., closer to zero-median) for `Volatility` than for the other two treatments. We also see in Table 2 that the developer is able to make a range estimate for all `Volatility` time series because the added Gaussian noise provides time-dense error information.

5.3 `Naive` obfuscates system shortcomings

Given proper visibility into the operation of the sensor system, we would expect user confidence and user satisfaction to correlate roughly with the fidelity of the system (i.e., be inversely correlated with error threshold). In Figure 4, we see this is true for `History` and `Volatility`, but not for `Naive`. In fact, at error thresholds of 1.5°F, average confidence and satisfaction both noticeably increase for `Naive`. Thus, `Naive` does not provide users with sufficient visibility into the sensing system, especially under high error thresholds.

	Treatment	Naive				History				Volatility			
	Error Threshold	0.0	0.5	1.0	1.5	0.0	0.5	1.0	1.5	0.0	0.5	1.0	1.5
Users	Correct direction (%)	73	51	33	31	87	78	79	56	75	50	**56**	**45**
	No change predicted (%)	17	38	52	60	11	17	19	36	17	35	28	41
	Incorrect direction (%)	10	11	15	9	2	5	2	9	8	15	16	14
Developer	Correct direction (%)	78	67	33	22	100	89	78	56	67	78	33	22
	No change predicted (%)	22	22	67	78	0	11	22	44	33	22	67	78
	Incorrect direction (%)	0	11	0	0	0	0	0	0	0	0	0	0

Table 3: Predicted trends in temperature for users and developer, sorted into "correct direction" (i.e., consistent with ground truth), "no change predicted," and "incorrect direction." `Volatility` causes users to predict nonzero trends more often than they do for `Naive`, indicating that `Volatility` fosters an understanding that the underlying temperature may be changing even if the displayed value remains constant. Users make these additional predictions heuristically, resulting in better-than-random predictions when compared with developer responses.

5.4 Participant Composition

Participants consisted of UC Berkeley students and staff registered in the XLab participant pool. Participant ages ranged from 18 to 53, with median 21 and mode 19. Participant fields-of-study consisted of 56% STEM, 37% non-STEM, and 7% unclassified (comprising "undeclared," "staff," and empty responses), with several participants listing double-majors in both STEM and non-STEM areas. Participants generally reported low familiarity with sensors, while the perceived difficulty of collecting temperature data across campus was independent of technical familiarity. We did not ask for participant gender.

6 FUTURE WORK

Additional work could continue exploring different task-based conceptual models and the subsequent system design; how to change the heuristics users use when interpreting data; and how to best align those heuristics with reality.

This work may also inspire the design of conceptual models and sensing systems that not only *describe* the real world, but also enable users to *experiment* with and *learn* about their surroundings. For example, in household power monitoring, a user might try flicking on and off a set of lights to see how much power they consume. Conceptually, one could imagine each household appliance drawing some fixed amount of power, such that total household power is a summation of individual components, which may appear as "blocks" or steps in a time series visualization of the data. An error-triggered data update strategy would not be able to support this conceptual model, because power data tend to have large transients that are much larger than e.g., steady-state power draw of a set of lights. Instead, data could be uploaded when a noticeable change in the mean power draw has occurred. This would result in a more interactive and educational system than a naive error-triggered implementation would.

7 PRACTICAL IMPLICATIONS

For low-fidelity data streams, data-reduction decisions made at the source heavily limit data visualization options at the sink and introduce statistical artifacts into the data that are misinterpreted by general users. The specific data features that are useful and/or misinterpreted are application- and task-specific. Thus, developers must do the due diligence of conducting user task analyses; constructing simple, task-based conceptual models; designing data-upload and power-management strategies that support intuitive visualization and user interaction of these models; conducting user studies to validate the system; and fighting the urge to introduce an endless slew of new features and tools. A great overview of this process can be found in [15].

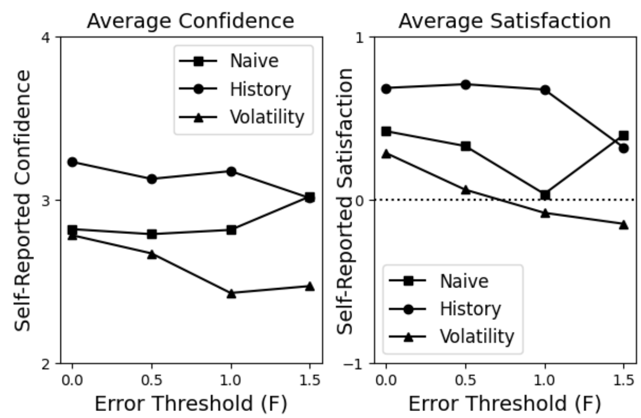


Figure 4: Average user confidence that their "answers are good descriptions of real-world circumstances," and average user satisfaction of having "this quality of data to use." `Naive` obfuscates system shortcomings from users, resulting in inappropriately high confidence and satisfaction when using a 1.5°F error threshold.

8 CONCLUSIONS

This work provides a "proof of existence" for the misinterpretation of low-fidelity data streams by general users, presents methods for articulating and quantifying these misinterpretations, and reveals insights and open questions for how developers could improve the truthfulness and usefulness of their sensing systems through task-based system design.

9 ACKNOWLEDGMENTS

This research was supported by the Experimental Social Science Laboratory (Xlab) at the University of California, Berkeley. This material is based upon work supported by the U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy (EERE) under the award number DE-EE0008220.

REFERENCES

[1] 2023. Amazon Sidewalk Privacy and Security Whitepaper. https://m.media-amazon.com/images/G/01/sidewalk/final_privacy_security_whitepaper.pdf.

[2] 2025. Find My Network. https://developer.apple.com/find-my/.

[3] 2025. Swarm Starlink. https://swarm.space/.

[4] Massimo Alioto. 2021. From Less Batteries to Battery-Less Alert Systems with Wide Power Adaptation down to nWs—Toward a Smarter, Greener World. *IEEE Design & Test* 38, 5 (2021), 90–133. https://doi.org/10.1109/MDAT.2021.3069087

[5] Rahul Basole, Elliot Bendoly, Aravind Chandrasekaran, and Kevin Linderman. 2021. Visualization in Operations Management Research. *INFORMS Journal on Data Science* 1, 2 (2021), 172–187. https://doi.org/10.1287/ijds.2021.0005

[6] Alex Bellon, Alex Yen, and Pat Pannuto. 2023. TagAlong: Free, Wide-Area Data-Muling and Services. In *Proceedings of the 24th International Workshop on Mobile Computing Systems and Applications*.

[7] Daniel Braun, Remco Chang, Michael Gleicher, and Tatiana von Landesberger. 2025. Beware of Validation by Eye: Visual Validation of Linear Trends in Scatterplots. *IEEE Transactions on Visualization and Computer Graphics* 31, 1 (Jan. 2025), 787–797. https://doi.org/10.1109/tvcg.2024.3456305

[8] Silvia Croce and Stefano Tondini. 2022. Fixed and Mobile Low-Cost Sensing Approaches for Microclimate Monitoring in Urban Areas: A Preliminary Study in the City of Bolzano (Italy). *Smart Cities* 5, 1 (2022), 54–70. https://doi.org/10.3390/smartcities5010004

[9] Karin Eberhard. 2023. The effects of visualization on judgment and decision-making: a systematic literature review. *Management Review Quarterly* 73 (2023), 167–214. https://doi.org/10.1007/s11301-021-00235-8

[10] Roberto Fabrizi, Stefania Bonafoni, and Riccardo Biondi. 2010. Satellite and Ground-Based Sensors for the Urban Heat Island Analysis in the City of Rome. *Remote Sensing* 2, 5 (2010), 1400–1415. https://doi.org/10.3390/rs2051400

[11] Amir Haleem, Andrew Allen, Andrew Thompson, Marc Nijdam, and Rahul Garg. 2018. Helium: A Decentralized Wireless Network. (2018).

[12] Gerhard P. Hancke, Bruno De Carvalho e Silva, and Gerhard P. Hancke, Jr. 2013. The Role of Advanced Sensing in Smart Cities. *Sensors* 13, 1 (2013), 393–425. https://doi.org/10.3390/s130100393

[13] Josiah Hester and Jacob Sorber. 2017. The Future of Sensing is Batteryless, Intermittent, and Awesome. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems* (Delft, Netherlands) *(SenSys '17)*. Association for Computing Machinery, New York, NY, USA, Article 21, 6 pages. https://doi.org/10.1145/3131672.3131699

[14] Jeff Johnson. 2014. Chapter 11 - Many Factors Affect Learning. In *Designing with the Mind in Mind (Second Edition)* (second edition ed.), Jeff Johnson (Ed.). Morgan Kaufmann, Boston, 149–167. https://doi.org/10.1016/B978-0-12-407914-4.00011-7

[15] Jeff Johnson and Austin Henderson. 2002. Conceptual models: begin by designing what to design. *Interactions* 9, 1 (Jan. 2002), 25–32. https://doi.org/10.1145/503355.503366

[16] Albert Kim, Eric Blais, Aditya Parameswaran, Piotr Indyk, Sam Madden, and Ronitt Rubinfeld. 2015. Rapid sampling for visualizations with ordering guarantees. *Proc. VLDB Endow.* 8, 5 (jan 2015), 521–532. https://doi.org/10.14778/2735479.2735485

[17] Shuai Liu, Yingcong Huang, Huoxiang Yang, Yongsheng Liang, and Wei Liu. 2022. End-to-end image compression method based on perception metric. *Signal, Image and Video Processing* 16, 7 (Oct. 2022), 1803–1810. https://doi.org/10.1007/s11760-022-02137-y

[18] Dominik Moritz, Danyel Fisher, Bolin Ding, and Chi Wang. 2017. Trust, but Verify: Optimistic Visualizations of Approximate Queries for Exploring Big Data. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing Machinery, New York, NY, USA, 2904–2915. https://doi.org/10.1145/3025453.3025456

[19] Carlos Enrique Muniz-Cuza, Soren Kejser Jensen, Jonas Brusokas, Nguyen Ho, and Torben Back Pederson. 2024. Evaluating the Impact of Error-Bounded Lossy Compression on Time Series Forecasting. In *Proceedings of the 27th International Conference on Extending Database Technology (EDBT)*. 650–663. https://doi.org/10.48786/edbt.2024.56

[20] Vinh T Nguyen, Kwanghee Jung, and Vibhuti Gupta. 2021. Examining data visualization pitfalls in scientific publications. *Visual Computing for Industry, Biomedicine, and Art* 4, 1 (2021). https://doi.org/10.1186/s42492-021-00092-y

[21] Ameya Patil, Gaelle Richer, Christopher Jermaine, Dominik Moritz, and Jean-Daniel Fekete. 2023. Studying Early Decision Making with Progressive Bar Charts. *IEEE Trans Vis Comput Graph* 29, 1 (Jan. 2023), 407–417. https://doi.org/10.1109/TVCG.2022.3209426

[22] Sajjadur Rahman, Maryam Aliakbarpour, Ha Kyung Kong, Eric Blais, Karrie Karahalios, Aditya Parameswaran, and Ronitt Rubinfield. 2017. I've seen "enough": incrementally improving visualizations to support rapid decision making. *Proc. VLDB Endow.* 10, 11 (aug 2017), 1262–1273. https://doi.org/10.14778/3137628.3137637

[23] Jean-Luc Watson, Tess Despres, Alvin Tan, Shishir G. Patil, Prabal Dutta, and Raluca Ada Popa. 2024. Nebula: A Privacy-First Platform for Data Backhaul. Cryptology ePrint Archive, Paper 2024/409. https://doi.org/10.1109/SP54263.2024.00092

[24] Berkeley Lab Weather. 2024. Weather Data at Berkeley Lab. https://sites.google.com/a/lbl.gov/lbnl-meteorological-data/lbnl-weather-data. [Online; accessed 8-May-2024].

[25] Yuefeng Zhang, Chuanmin Jia, Jiannhui Chang, and Siwei Ma. 2023. Machine Perception-Driven Image Compression: A Layered Generative Approach. arXiv:2304.06896 [eess.IV]