Three principles of data science: predictability, computability, and stability (PCS)

Bin Yu[a,b,c] **and Karl Kumbier**[a]

[a]Statistics Department, University of California, Berkeley, CA 94720; [b]EECS Department, University of California, Berkeley, CA 94720; [c]Chan Zuckerberg Biohub, San Francisco, CA 94158

Drawing from three principles of data science: predictability, computability, and stability (PCS), we propose the PCS framework to provide responsible, reliable, reproducible, and transparent results across the entire data science life cycle. It uses predictability as a reality check and considers the importance of computation in data collection/storage and algorithm design. We augment predictability and computability with an overarching stability principle. Stability expands on statistical uncertainty considerations to assess how human judgment calls impact data results through data and model/algorithm perturbations.

We develop inference procedures that build on PCS, namely PCS perturbation intervals and PCS hypothesis testing, to investigate the stability of data results relative to problem formulation, data cleaning, and modeling decisions. We compare PCS inference with existing methods in high-dimensional sparse linear model simulations to demonstrate that our approach compares favorably to others in terms of ROC curves over a wide range of simulation settings. Finally, we propose PCS documentation based on R Markdown or Jupyter Notebook, with publicly available, reproducible codes and narratives to back up human choices made throughout an analysis. The PCS workflow and documentation are demonstrated in a genomics case study available on Zenodo.

1. Introduction

Data science is a field of evidence seeking that combines data with domain information to generate new knowledge. The data science life cycle (DSLC) begins with a domain question or problem and proceeds through collecting, managing, processing/cleaning, exploring, modeling, and interpreting* data results to guide new actions (Fig. 1). Given the transdisciplinary nature of this process, data science requires human involvement from those who understand both the data domain and the tools used to collect, process, and model data. These individuals make implicit and explicit judgment calls throughout the DSLC. The limited transparency in reporting such judgment calls has blurred the evidence for many analyses, resulting in more false-discoveries than might otherwise occur (2, 3). In order to mitigate this problem, data science requires an enriched technical language for clearly communicating and evaluating empirical evidence in the context of human decisions. Three core principles: predictability, computability, and stability (PCS) provide the foundation for such a data-driven language. They serve as minimum requirements for extracting reliable, reproducible information from data.

PCS have been widely used across various areas of data science. Predictability plays a central role in science through Popperian falsifiability (4). If a model does not accurately predict new observations, it can be rejected or updated. Predictability has been adopted by the machine learning community as a goal of its own right and more generally to evaluate the quality of a model or data result (5). While statistics has always considered prediction, machine learning emphasized its importance for empirical rigor. This was in large part powered by computational advances that made it possible to compare models through cross-validation (CV), a technique developed by statisticians Stone and Allen (6, 7).

The role of computation extends beyond prediction, setting limitations on how data can be collected, stored, and analyzed. Computability has played an integral role in computer science tracing back to Alan Turing's seminal work on the computability of sequences (8). Analyses of computational complexity have since been used to evaluate the tractability of machine learning algorithms (9). Kolmogorov built on Turing's work through the notion of Kolmogorov complexity, which describes the minimum computational resources required to represent an object (10, 11). Since Turing machine-based notions of computabiltiy are not computable in practice, we treat computability as an issue of algorithm efficiency and scalability. This narrow definition of computability addresses computational considerations at the modeling stage of the DSLC but does not deal with data collection, storage, or cleaning.

Stability† is a common sense principle and a prequisite for knowledge. It is related to the notion of scientific reproducibility, which Fisher and Popper argued is a necessary condition for establishing scientific results (4, 12). While replicability across laboratories has long been an important consideration in science, computational reproducibility has come to play an important role in data science as well. For example, (13) discusses reproducible research in the context of computational harmonic analysis. More broadly, (14) advocates for "preproducibility" to explicitly detail all steps along the DSLC and ensure sufficient information for quality control. Stability at

---

*For a precise definition of interpretability in the context of machine learning, we refer to our recent paper (1)

†We differentiate between the notions of stability and robustness as used in statistics. The latter has traditionally been used to investigate performance of statistical methods across a range of distributions, while the former captures a much broader range of perturbations throughout the DSLC as discussed in this paper. At a high level, stability is about robustness.
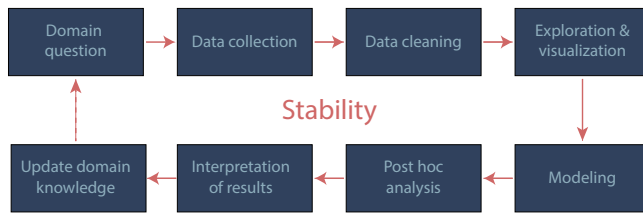
Fig. 1. The data science life cycle

the modeling stage of the DSLC has been advocated in (15) as a minimum requirement for reproducibility and interpretability. Modeling stage stability unifies numerous previous works, including Jackknife, subsampling, bootstrap sampling, robust statistics, semi-parametric statistics, and Bayesian sensitivity analysis (see (15) and references therein). These methods have been enabled in practice through computational advances and allow researchers to investigate the reproducibility of data results. Econometric models with partial identification (see the book (16) and references therein) and fundamental theoretical results in statistics, such as the central limit theorem (CLT), can also be viewed as stability considerations.

In this paper, we unify and expand on these ideas through the PCS framework, which integrates PCS-based inference procedures with transparent documentation. It uses predictability as a reality check, computability to ensure that the DSLC is tractable, and stability to test the reproducibility of data results (Sec. 2). We develop PCS inference, which leverages data and model perturbations to evaluate the uncertainty human decisions introduce into the DSLC (Sec. 3). We propose PCS documentation in R MarkDown or a Jupyter (iPython) Notebook to justify these decisions through narratives, code and visualizations (Sec. 4). We draw connections between causal inference and the PCS framework, demonstrating the utility of the latter as a recommendation system for generating scientific hypotheses (Sec. 5). We conclude by discussing areas for further work, including additional vetting of the framework and theoretical analyses on the connections between the three principles. A case study of our proposed framework based on the authors' work studying gene regulation in *Drosophila* is documented on Zenodo.

2. PCS principles in the DSLC

Given a domain problem and data, the purpose of the DSLC is to generate knowledge, conclusions, and/or actions (Fig. 1). The PCS framework aims to ensure that this process is both reliable and reproducible through three fundamental principles of data science. Below we discuss the three principles and formulate their roles within the PCS framework.[‡] We address PCS documentation in Sec. 4.

A. Stability assumptions initiate the DSLC. The ultimate goal of the DSLC is to generate knowledge that is useful for future actions, be it a biological experiment, business decision, or government policy. Stability is a useful concept to address whether another researcher making alternative, appropriate[§]

[‡]We organize our discussion with respect to the DSLC as opposed to the name PCS, which originated earlier from talks given by the authors.

[§]We use the term appropriate to mean well-justified from domain knowledge and an understanding of the data generating process. The term "reasonable" has also been used with this definition (15).

decisions would obtain similar conclusions. At the modeling stage, stability has previously been advocated in (15). In this context, stability refers to acceptable consistency of a data result relative to appropriate perturbations of the data or model. For example, jackknife (17–19), bootstrap (20), and cross validation (6, 7) may be considered appropriate perturbations if the data are deemed approximately independent and identically distributed (i.i.d.) based on domain knowledge and an understanding of the data collection process.

Human judgment calls prior to modeling also impact data results. The validity of an analysis relies on implicit stability assumptions that allow data to be treated as an accurate representation of some natural phenomena. When these assumptions do not hold, conclusions do not generalize to new data and should rarely be used to guide future actions. This makes it essential to evaluate stability to guard against costly future actions and false discoveries, particularly in the domains of science, business, and public policy, where data results are used to guide large scale actions, and in medicine, where human lives are at stake. Below we outline stability considerations that impact the DSLC prior to modeling.

Question or problem formulation: The DSLC begins with a domain problem or a question, which could be hypothesis-driven or discovery-based. For instance, a biologist may want to discover biomolecules that regulate a gene's expression. In the DSLC this question must be translated into a question regarding the output of a model or analysis of data that can be measured/collected. There are often multiple translations of a domain problem into a data science problem. For example, the biologist described above could measure factors binding regulatory regions of the DNA that are associated with the gene of interest. Alternatively, she could study how the gene covaries with regulatory factors across time and/or space. From a modeling perspective, the biologist could identify important features in a random forest or through logistic regression. Stability relative to question or problem formulation implies that the domain conclusions are qualitatively consistent across these different translations.

Data collection: To answer a domain question, domain experts and data scientists collect data based on prior knowledge and available resources. When this data is used to guide future decisions, researchers implicitly assume that the data is relevant for a future time and under future conditions. In other words, they assume that conditions affecting data collection are stable, at least relative to some aspects of the data. For instance, if multiple laboratories collect data to answer a domain question, protocols must be comparable across experiments and laboratories if they expect to obtain consistent results. These stability considerations are closely related to external validity in medical research, which characterizes similarities between subjects in a study and subjects that researchers hope to generalize results to. We will discuss this idea more in Sec. B.

Data cleaning and preprocessing: Statistics and machine learning models or algorithms help data scientists answer domain questions. Using these models or algorithms requires cleaning and/or processing raw data into a suitable format, be it a categorical demographic feature or continuous measurements of biomarker concentrations. For instance, when data come from multiple laboratories, biologists must decide how to normalize individual measurements (for example see (21)).

When data scientists preprocess data, they are implicitly assuming that their choices are not unintentionally biasing the essential information in the raw data. In other words, they assume that the knowledge derived from a data result is stable with respect to their processing choices. If such an assumption cannot be justified, they should use multiple appropriate processing methods and interpret data results that are stable across these methods. Others have advocated evaluating results across alternatively processed datasets under the name "multiverse analysis" (22). Although the stability principle was developed independently of this work, it naturally leads to a multiverse-style analysis.

Exploratory data analysis: Both before the modeling stage and in post hoc analyses, data scientists often engage in exploratory data analysis (EDA) to identify interesting relationships in the data and interpret data results. When visualizations or summaries are used to communicate these analyses, it is implicitly assumed that the relationships or data results are stable with respect to any decisions made by the data scientist. For example, if the biologist believes that clusters in a heatmap represent biologically meaningful groups, she should expect to observe the same clusters with respect to any appropriate choice of distance metric, data perturbation, or clustering method.

B. Predictability as reality check. After data collection, cleaning/preprocessing, and EDA, models or algorithms[¶] are frequently used to identify more complex relationships in data. Many essential components of the modeling stage rely on the language of mathematics, both in technical papers and in code. A seemingly obvious but often ignored question is why conclusions presented in the language of mathematics depict reality that exists independently in nature, and to what extent we should trust mathematical conclusions to impact this external reality.[ǁ]

This concern has been articulated and addressed by many others in terms of prediction. For instance, Philip Dawid drew connections between statistical inference and prediction under the name "prequential statistics," highlighting the importance of forecasts in statistical analyses (23). David Freedman argued that when a model's predictions are not tested against reality, conclusions drawn from the model are unreliable (24). Seymour Geisser advocated that statistical analyses should focus on prediction rather than parametric inference, particularly in cases where the statistical model is an inappropriate description of reality (25). Leo Breiman championed the essential role of prediction in developing realistic models that yield sound scientific conclusions (5). It can even be argued that the goal of most domain problems is prediction at the meta level. That is, the primary value of learning relationships in data is often to predict some aspect of future reality.

B.1. Formulating prediction. We describe a general framework for prediction with data $D = (\mathbf{x}, y)$, where $\mathbf{x} \in \mathcal{X}$ represents input features and $y \in \mathcal{Y}$ the prediction target. Prediction targets $y \in \mathcal{Y}$ may be observed responses (e.g. supervised learning) or extracted from data (e.g. unsupervised learning). Predictive accuracy provides a simple, quantitative metric to evaluate how well a model represents relationships in D. It is well-defined relative to a prediction function, testing data,

and an evaluation function. We detail each of these elements below.

Prediction function: The prediction function

$$h : \mathcal{X} \to \mathcal{Y} \qquad [1]$$

represents relationships between the observed features and the prediction target. For instance, in the case of supervised learning h may be a linear predictor or decision tree. In this setting, y is typically an observed response, such as a class label. In the case of unsupervised learning, h could map from input features to cluster centroids.

To compare multiple prediction functions, we consider

$$\{h^{(\lambda)} : \lambda \in \Lambda\}, \qquad [2]$$

where Λ denotes a collection models/algorithms. For example, Λ may define different tuning parameters in lasso (26) or random forest (27). For deep nerual networks, Λ could describe different network architectures. For algorithms with a randomized component, such as k-means or stochastic gradient descent, Λ can represent repeated runs. More broadly, Λ may describe a set of competing algorithms such as linear models, random forests, and neural networks, each corresponding to a different problem translations. We discuss model perturbations in more detail in Sec. D.3.

Testing (held-out) data: We distinguish between *training data* that are used to fit a collection of prediction functions, and held-out *testing data* that are used to evaluate the accuracy of fitted prediction functions.[**] *Internal testing* data are collected at the same time and under the same conditions as the training data. At a minimum, one should evaluate predictive accuracy on internal testing data. To evaluate how a model performs in future settings, one may also consider *external testing* data gathered under different conditions from the training data. Prediction accuracy on external testing data directly addresses questions of external validity, which describe how well a result will generalize to future observations. Domain knowledge from humans involved in data generation and analysis is essential to assess the appropriateness of different prediction settings. These decisions should be reported in the proposed PCS documentation (Sec. 4).

Prediction evaluation metric: The prediction evaluation metric

$$\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_+ \qquad [3]$$

quantifies the accuracy of a prediction function $h \in \mathcal{H}$ by measuring the similarity between $h(\mathbf{x})$ and y. We adopt the convention that increasing values of $\ell(h, \mathbf{x}, y)$ imply worse predictive accuracy. The prediction evaluation metric should be selected to reflect domain-specific considerations, such as the types of errors that are more costly. In fact, there is an entire area of research devoted to evaluating the quality of probabilistic forecasts through "scoring rules" (see (28) and references therein). In some cases, it may be appropriate to consider multiple prediction evaluation metrics and focus on models that are deemed accurate with respect to all.

Prediction requires human input to formulate, including the preferred structure of a model/algorithm and what it means for a model to be suitably accurate. For example, the biologist

---

[¶] The model or algorithm choices could correspond to different translations of a domain problem.

[ǁ] The PCS documentation in Sec. 4 helps users assess whether this connection is reliable.

[**] In some settings, a third set of validation data are used to tune model parameters

studying gene regulation may believe that the simple rules learned by decision trees are an an appealing representation of interactions that exhibit thresholding behavior (29). If she is interested in a particular cell-type, she may evaluate prediction accuracy on internal test data measuring only these environments. If her responses are binary with a large proportion of class-0 responses, she may choose an evaluation function ℓ to handle the class imbalance. All of these decisions should be documented and argued for by researchers (Sec. 4) so that others can review and assess the strength of conclusions based on transparent evidence. The accompanying PCS documentation provides a detailed example.

B.2. Cross validation. As alluded to earlier, CV has become a powerful work horse to select regularization parameters when data are approximately i.i.d. (6, 7). CV divides data into blocks of observations, trains a model on all but one block, and evaluates the prediction error over each held-out block. That is, CV incorporates the scientific principle of replication by evaluating whether a model accurately predicts the responses of pseudo-replicates. CV works more effectively as a tool to select regularization parameters than as an estimate of prediction error, where it can incur high variability due to the often positive dependences between the estimated prediction errors in the summation of the CV error (30). Just as peer reviewers make judgment calls on whether a lab's experimental conditions are suitable to replicate scientific results, data scientists must determine whether a removed block represents a justifiable pseudo replicate of the data, which requires information from the data collection process and domain knowledge.

C. Computability. In a broad sense, computability is the gatekeeper of data science. If data cannot be generated, stored, managed, and analyzed efficiently and scalably, there is no data science. Modern science relies heavily on information technology as part of the DSLC. Each step, from raw data collection and cleaning, to model building and evaluation, rely on computing technology and fall under computability in a broad sense. In a narrow sense, computability refers to the computational feasibility of algorithms or model building.

Here we use computability in the narrow-sense, which is closely associated with the rise of machine learning over the last three decades. Just as scientific instruments and technologies determine what processes can be effectively measured, computing resources and technologies determine the types of analyses that can be carried out. In particular, computability is necessary to carry out predictability and stability analyses within the PCS framework. Computational constraints can also serve as a device for regularization. For example, stochastic gradient descent is widely used for optimization in machine learning problems (31). Both the stochasticity and early stopping of a stochastic gradient algorithm play the role of implicit regularization.

Computational considerations and algorithmic analyses have long been an important part of statistics and machine learning. Even before digital computing, calculus played a computational role in statistics through Taylor expansions applied to different models. In machine learning, computational analyses consider the number of operations and required storage space in terms of observations n, features p, and tuning (hyper) parameters. When the computational cost of addressing a domain problem or question exceeds available computational resources, a result is not computable. For instance, the biologist interested in gene regulation may want to model interaction effects in a supervised learning setting. However, there are $O(p^s)$ possible order-s interactions among p regulatory factors. For even a moderate number of factors, exhaustively searching for high-order interactions is not computable. In such settings, data scientists must restrict modeling decisions to draw conclusions. Thus it is important to document why certain restrictions were deemed appropriate and the impact they may have on conclusions (Sec. 4).

Increases in computing power also provide an unprecedented opportunity to enhance analytical insights into complex natural phenomena. We can now store and process massive datasets and use these data to simulate large scale processes. Simulations provide concrete and quantitative representations of a natural phenomena relative to known input parameters, which can be perturbed to assess the stability of data results. As a result, simulation experiments inspired by observed data and domain knowledge are powerful tools to understand how results may behave in real-world settings. They represent a best effort to emulate complex processes, where the reliability of data results is not always clear. Pairing such simulation studies with empirical evidence makes the DSLC more transparent for peers and users to review, aiding in the objectivity of science.

D. Stability at the modeling stage. Computational advances have fueled our ability to analyze the stability of data results in practice. At the modeling stage, stability measures how a data result changes when the data and/or model are perturbed. Stability extends the concept of sampling variability in statistics, which is a measure of instability relative to other data that could be generated from the same distribution. Statistical uncertainty assessments implicitly assume stability in the form of a distribution that generated the data. This assumption highlights the importance of other data sets that could be observed under similar conditions (e.g. by another person in the lab or another lab at another time).

The concept of a true distribution is a construct. When randomization is explicitly carried out, the true distribution can be viewed as a physical construct. Otherwise, it is a mental construct that must be justified through domain knowledge, an understanding of the data generating process, and downstream utility. Statistical inference procedures use distributions to draw conclusions about the real world. The relevance of such conclusions requires empirical support for the postulated true distribution, especially when it is a mental construct. In data science and statistical problems, practitioners often do not make much of an attempt to justify this mental construct. At the same time, they take the uncertainty conclusions very seriously. This flawed practice is likely related to the high rate of false discoveries (2, 3). It is a major impediment to progress of science and to data-driven knowledge extraction in general.

While the stability principle encapsulates uncertainty quantification (when the true distribution construct is well supported), it is intended to cover a much broader range of perturbations, such as problem formulation (e.g. different problem translations), pre-processing, EDA, randomized algorithms, and choice of models/algorithms. Although seldom carried out in practice, evaluating stability across the entire DSLC is necessary to ensure that results are reliable and reproducible. For example, the biologist studying gene regulation must choose

both how to normalize raw data and what algorithm(s) she will use in her analysis. When there is no principled approach to make these decisions, the knowledge data scientists can extract from analyses is limited to conclusions that are stable across appropriate choices (22, 32, 33). This ensures that another scientist studying the same data will reach similar conclusions, despite slight variation in their independent choices.

D.1. Formulating stability at the modeling stage.
Stability at the modeling stage is defined with respect to a target of interest, an appropriate perturbation to the data and/or algorithm/model, and a stability metric to measure the change in target that results from perturbation. We describe each of these in detail below.

Stability target: The stability target

$$\mathcal{T}(D, \lambda), \tag{4}$$

corresponds to the data result or estimand of interest. It depends on input data D and a specific model/algorithm λ used to analyze the data. For simplicity, we will sometimes suppress the dependence on D and λ in our notation. As an example, $\mathcal{T}$ can represent responses predicted by $h^{(\lambda)}$. Other examples of $\mathcal{T}$ include features selected by lasso with penalty parameter λ or saliency maps derived from a convolutional neural network (CNN) with architecture λ.

Data and model/algorithm perturbations: To evaluate the stability of a data result, we measure the change in target $\mathcal{T}$ that results from a perturbation to the input data or learning algorithm. More precisely, we define a collection of data perturbations $\mathbf{D}$ and model/algorithm perturbations Λ and compute the stability target distribution

$$\{\mathcal{T}(D, \lambda) : D \in \mathbf{D}, \lambda \in \Lambda\}. \tag{5}$$

For example, appropriate data perturbations include bootstrap sampling when observations are approximately i.i.d., block bootstrap for weakly dependent time series, generative models that are supported by domain knowledge (Sec. D.2), and probabilistic models that are justified from an understanding of the data generating process or explicit randomization. When different prediction functions are deemed equally appropriate based on domain knowledge, each may represent an appropriate model perturbation (Sec. D.3).

It can be argued that the subjectivity surrounding appropriate perturbations makes it difficult to evaluate results within the PCS framework. Indeed, perturbation choices are both subjective human judgment calls and critical considerations of PCS. The degree to which a data result can be trusted depends on the justification for a perturbation. This is true if the perturbation comes from a probabilistic model, as in traditional statistical inference, or some broader set of perturbations, as in PCS. The goal of PCS is to use and explicitly document perturbations that are best suited to assess stability in complex, high-dimensional data rather than relying on probabilistic models alone, which have little objective meaning when the model is not justified. To ensure that results can be evaluated, the case for an appropriate perturbation must be made in the publication and in the PCS documentation (Sec. 4). These transparent narratives allow readers to scrutinize and discuss perturbations to determine which should be applied for a particular field and/or type of data, aiding the objectivity of science.

Stability evaluation metric: The stability evaluation metric $s(\mathcal{T}; \mathbf{D}, \Lambda)$ summarizes the stability target distribution in Eq. (5). For example, if $\mathcal{T}$ indicates features selected by a model trained on data D, we may report the proportion of times each feature is selected across data perturbations $D \in \mathbf{D}$. If $\mathcal{T}$ corresponds to saliency maps derived from different CNN architectures $\lambda \in \Lambda$, we may report each pixel's range of salience across Λ. When the stability evaluation metric combines targets across model/algorithm perturbations, it is important that these different targets are scaled appropriately to ensure comparability.

A stability analysis that reveals the target $\mathcal{T}$ is unstable (relative to a meaningful threshold for a particular domain) may suggest an alternative analysis or target of interest. This raises issues of multiplicity and/or overfitting if the same data are used to evaluate new stability targets. Held-out test data offer one way to mitigate these concerns. That is, training data can be used to identify a collection of targets that are suitably stable. These targets can then be evaluated on the test data. More broadly, the process of refining analyses and stability targets can be viewed as part of the iterative approach to data analysis and knowledge generation described by (34). Before defining a new target or analysis, it may be necessary to collect new data to help ensure reproducibility.

D.2. Data perturbation.
The goal of data perturbation under the stability principle is to mimic a process that could have been used to produce final input data but was not. This includes human decisions, such as preprocessing and data cleaning, as well as data generating mechanisms. When we focus on the change in target under possible realizations of the data from a well-supported probabilistic model, we arrive at well-justified sampling variability considerations in statistics. Hence data perturbation under the stability principle includes, but is much broader than, the concept of sampling variability. It formally recognizes many other important considerations in the DSLC beyond sampling variability. Furthermore, it provides a framework to build confidence measures for estimates of $\mathcal{T}$ when a probabilistic model is not well-justified and hence sampling interpretations are not applicable.

Data perturbations can also be used to reduce variability in the estimated target, which corresponds to a data result of interest. Random forests incorporate subsampling data perturbations (of both the data units and predictors) to produce predictions with better generalization error (27). Generative adversarial networks (GANs) use synthetic adversarial examples to re-train deep neural networks and produce predictions that are more robust to such adversarial data points (35). Bayesian models based on conjugate priors lead to marginal distributions that can be derived by adding observations to the original data. Thus they can be viewed as a form of data perturbation that implicitly introduces synthetic data through the prior. Empirically supported generative models, including PDEs, can be used to explicitly introduce synthetic data. As with Bayesian priors, synthetic data perturbations from generative models can be used to encourages stability of data results relative to prior knowledge, such as mechanistic rules based on domain knowledge (for examples see (36)).

D.3. Algorithm or model perturbation.
The goal of algorithm or model perturbation is to understand how alternative analyses of the same data affect the target estimate. A classical example

of model perturbation is from robust statistics, where one searches for a robust estimator of the mean of a location family by considering alternative models with heavier tails than the Gaussian model. Another example of model perturbation is sensitivity analysis in Bayesian modeling (37, 38). Many of the model conditions used in causal inference are in fact stability concepts that assume away confounding factors by asserting that different conditional distributions are the same (39, 40).

Modern algorithms often have a random component, such as random projections or random initial values in gradient descent and stochastic gradient descent. These random components provide natural model perturbations that can be used to assess the stability of $\mathcal{T}$. In addition to the random components of a single algorithm, multiple models/algorithms can be used to evaluate stability of the target. This is useful when there are many appropriate choices of model/algorithm and no established criteria or established domain knowledge to select among them. The stability principle calls for interpreting only the targets of interest that are stable across these choices of algorithms or models (32).

As with data perturbations, model perturbations can help reduce variability or instability in the target. For instance, (41) selects lasso coefficients that are stable across different regularization parameters. Dropout in neural networks is a form of algorithm perturbation that leverages stability to improve generalizability (42). Our previous work (33) stabilizes random forests to interpret the decision rules in an ensemble of decision trees (33, 43), which are perturbed using random feature selection (model perturbation) and bootstrap (data perturbation).

E. Dual roles of generative models in PCS. Generative models include both probabilistic models and partial differential equations (PDEs) with initial or boundary conditions. These models play dual roles in the PCS framework. On one hand, they can concisely summarize past data and prior knowledge. On the other hand, they can be used to generate synthetic observations that offer a form of data perturbation.

When a generative model is used to summarize data, a common target of interest is the model's parameters. Generative models with known parameters may be used for prediction or to advance understanding through the mechanistic rules they represent. Such models correspond to infinite data, though finite under computational constraints. Generative models with unknown parameters can be used to motivate surrogate loss functions through maximum likelihood and Bayesian modeling methods. Mechanistic interpretations of such models should not be used to draw scientific conclusions. They are simply useful starting points to optimize algorithms that must be subjected to empirical validation.

Generative models that approximate the data generating process, a human judgment call argued for in the PCS documentation, can be used as a form of data perturbation. Here synthetic data augment the observed data and serve the purpose of domain-inspired regularization. The amount of synthetic data to combine with the observed data reflects our degree of belief in the models. Using synthetic data for domain inspired regularization allows the same algorithmic and computing platforms to be applied to the combined data. This style of analysis is reminiscent of AdaBoost, which use the current data and model to modify the data used in the next iteration without changing the base-learner (44).

F. Connections among the PCS principles. Although we have discussed the three principles of PCS individually, they share important connections. Computational considerations can limit the predictive models/algorithms that are tractable, particularly for large, high-dimensional datasets. These computability issues are often addressed in practice through scalable optimization methods such as gradient descent (GD) or stochastic gradient descent (SGD). Evaluating predictability on held-out data is a form of stability analysis where the training/test sample split represents a data perturbation. Other perturbations used to assess stability require multiple runs of similar analyses. Parallel computation is well suited for these perturbations.

3. PCS inference through perturbation analysis

When data results are used to guide future decisions or actions, it is important to assess the quality of the target estimate. For instance, suppose a model predicts that an investment will generate a 10% return over one year. Intuitively, this prediction suggests that "similar" investments return 10% on average. Whether or not a particular investment will realize a return close to 10% depends on whether returns for "similar" investments ranged from −20% to 40% or from 8% to 12%. In other words, the variability of a prediction conveys important information about how much one should trust it.

In traditional statistics, confidence measures describe the uncertainty of an estimate due to sampling variability under a well-justified probabilistic model. However, decisions made throughout the DSLC add another layer of uncertainty that may bias data results. This issue has been previously acknowledged in the modeling stage by (45), who derive "hacking intervals" to assess the range of a summary statistic optimized over a possible set of data and algorithm perturbations. In the PCS framework, we propose *perturbation intervals* to quantify the stability of target estimates relative to different perturbations, including data cleaning/preprocessing and problem translations. Perturbation intervals are conceptually similar to confidence intervals. The primary difference is that they are explicitly connected to perturbations, which are justified in PCS documentation (Sec. 4) and evaluated by independent reviewers and domain experts.

As an example, perturbation intervals for a single method based on bootstrap sampling specialize to traditional confidence intervals based on the bootstrap. More broadly, perturbation intervals quantify the variability of a target parameter across the entire DSLC. For instance, a data scientist may consider multiple preprocessing, subsampling, and modeling strategies to predict investment returns. The resulting perturbation intervals describe the range of returns across worlds represented by each perturbation. Their reliability lies squarely on whether the set of perturbations captures the full spectrum of appropriate choices that could be made throughout the DSLC, which should be evaluated by domain experts and independent reviewers. This highlights the importance of perturbations that could plausibly generate the observed data, represent the range of uncertainty surrounding an analysis to the best degree possible, and are transparently documented for others to evaluate (Sec. 4).

As a starting point, we focus on a basic form of PCS inference that generalizes traditional statistical inference. Our approach to inference allows for a range of data and algo-

rithm/model perturbations, making it flexible in its ability to represent uncertainty throughout the DSLC.

A. PCS perturbation intervals. The reliability of perturbation intervals lies on the appropriateness of each perturbation. Consequently, perturbation choices should be seriously deliberated, clearly communicated, and evaluated by objective reviewers. Here we propose a framework for PCS inference based on a single problem translation and target estimand, leaving the case of multiple translations/estimands to future work.[††]

1. **Problem formulation:** Translate the domain question into a data science problem that specifies how the question will be addressed. Define a prediction target y, appropriate data $\mathbf{D}$ and/or model Λ perturbations, prediction function(s) $\{h^{(\lambda)} : \lambda \in \Lambda\}$, training/test split, prediction evaluation metric ℓ, stability metric s, and stability target $\mathcal{T}(D, \lambda)$. Document why these choices are appropriate in the context of the domain question.

2. **Prediction screening:** For a threshold τ, screen out models that do not fit the data (as measured by prediction accuracy)

$$\Lambda^* = \{\lambda \in \Lambda : \ell(h^{(\lambda)}, \mathbf{x}, y) < \tau\}. \qquad [6]$$

Examples of appropriate threshold include domain accepted baselines, the top k performing models, or models whose accuracy is suitably similar to the most accurate model. If the goal of an analysis is prediction, testing data should be held-out until reporting the final prediction accuracy of a model in step 4. When test data cannot be used to screen models, Eq. (6) can be evaluated using a surrogate sample-splitting approach such as CV. If the goal of an analysis extends beyond prediction (e.g. to feature selection), Eq. (6) may be evaluated on held-out test data.

3. **Target value perturbation distributions:** For each of the survived models Λ^* from step 2, compute the stability target under each data perturbation $\mathbf{D}$. This results in a joint distribution of the target over data and model perturbations as in Eq. (5). For a collection of perturbations, requiring stability of $\mathcal{T}$ across all perturbations is more conservative in terms of type I error than requiring stability for any single perturbation. However, different domain questions require control over different types of error. How and when to combine results across perturbations is thus a human judgment call that should be transparently justified and documented.

4. **Perturbation result reporting:** Summarize the target value perturbation distribution using the stability metric s. For instance, if $\mathcal{T}$ is one-dimensional we could summarize its perturbation distribution using the 10th and 90th percentiles or a visualization. If $\mathcal{T}$ is multi-dimensional, we could report a low dimensional projection of the perturbation distribution. When perturbation results combine targets across models/algorithms, they may need to be rescaled to ensure comparability. When perturbation

intervals are reported separately for model/algorithm perturbation, predictive accuracy evaluated in step 2 may be used as a measure of trust to rank each interval.

At a high level, the PCS inference uses perturbation intervals to identify the stable part of accurate models. If perturbation results reveal instability among accurate models, PCS inference can be used to interpret aspects that are shared (i.e. stable) across these models. In this setting, PCS can be viewed as an implicit application of Occam's razor. That is, it draws conclusions from the stable portion of predictive models to simplify data results, making them more reliable and easier to interpret. If perturbation intervals reveal that complex models are both stable and accurate, PCS inference provides justification for the added complexity.

B. PCS hypothesis testing. Hypothesis testing from traditional statistics is commonly used in decision making for science and business alike. The heart of Fisherian testing (46) lies in calculating the p-value, which represents the probability of an event more extreme than in the observed data under a null hypothesis or distribution. Smaller p-values correspond to stronger evidence against the null hypothesis or (ideally) the scientific theory embedded in the null hypothesis. For example, we may want to determine whether a particular gene is differentially expressed between breast cancer patients and a control group. Given i.i.d. random samples from each population, we could address this question in the classical hypothesis testing framework using a t-test. The resulting p-value describes the probability of seeing a difference in means more extreme than observed if the genes are not differentially expressed.

While hypothesis testing is valid philosophically, many of the assumptions that it relies on are unrealistic in practice. For instance, unmeasured confounding variables can bias estimates of causal effects. These issues are particularly relevant in the social sciences, where randomized trials are difficult or impossible to conduct. Resource constraints can limit how data are collected, resulting in samples that do not reflect the population of interest, distorting the probabilistic interpretations of traditional statistical inference. Moreover, hypothesis testing assumes empirical validity of probabilistic data generating models.[‡‡] When randomization is not carried out explicitly, a particular null distribution must be justified from domain knowledge of the data generating mechanism. Such issues are seldom taken seriously in practice, resulting in settings where the null distribution is far from the observed data. As a result, p-values as small as 10^{-5} or 10^{-8} are now common to report, despite the fact that there are rarely enough data to reliably calculate these values, especially when multiple hypotheses (e.g. thousands of genes) are evaluated. When results are so far off on the tail of the null distribution, there is no empirical evidence as to why the tail should follow a particular parametric distribution. Moreover, hypothesis testing as practiced today often relies on analytical approximations or Monte Carlo methods, where issues arise for such small probability estimates. In fact, there is a specialized area of importance sampling to deal with simulating small probabilities (48, 49), but these ideas have not been widely adopted in practice.

[††] The PCS perturbation intervals cover different problem translations through Λ and are clearly extendable to include perturbations in the pre-processing step through $\mathbf{D}$.

[‡‡] Under conditions, Freedman (47) showed that some tests can be approximated by permutation tests when data are not generated from a probabilistic model, but these results are not broadly applicable.

PCS hypothesis testing builds on perturbation intervals to address these practical issues and the cognitively misleading nature of small p-values. It uses the null hypothesis to define constrained perturbations that represent a plausible data generating process, which in the best case corresponds to an existing scientific theory. This includes probabilistic models, when they are well founded, as well as other data and/or algorithm perturbations. For instance, generative models based on PDEs can be used to simulate data according to established physical laws. Alternatively, subsampling can be used to select control populations where some ground truth is known to hold (as an example see (50)). By allowing for a broad class of perturbations, PCS hypothesis testing allows us to compare observed data with data that respects some simple structure known to represent important characteristics of the domain question. Of course, the appropriateness of a perturbation is a human judgment call that should be clearly communicated in PCS documentation and debated by researchers. Much like scientists deliberate over appropriate controls in an experiment, data scientists should debate the appropriate perturbations in a PCS analysis.

B.1. Formalizing PCS hypothesis testing. Formally, we consider settings with observable input features $\mathbf{x} \in \mathcal{X}$, prediction target $y \in \mathcal{Y}$, prediction functions $\{h^{(\lambda)} : \lambda \in \Lambda\}$, and a null hypothesis that qualitatively describes some aspect of the domain question. PCS hypothesis testing translates the null hypothesis into a *constrained perturbation* and generates data

$$D_0 = \{\mathbf{x}_0, y_0\} \qquad [7]$$

according to this perturbation.[§§] The particular choice of constrained perturbation should be explicitly documented and justified by domain knowledge. We use the constrained perturbation to construct and compare perturbation intervals for both D_0 and D and evaluate whether the observed data is consistent with the embedded hypothesis.

C. PCS inference in neuroscience and biology. As an example of PCS hypothesis testing in practice, (51) considers the null hypothesis that population level structure in single neuron data is the expected byproduct of primary features (e.g. correlations across time). The authors use a maximum entropy approach, whose constraint is represented by the number of moments, to generate data that share primary features with the observed data, but are otherwise random, and compare population level findings between the observed and simulated data. In the accompanying PCS documentation, we consider the null hypothesis that genomic interactions appear with equal frequency among different classes of genomic elements. We use a sample splitting strategy which treats inactive elements (class-0 observations) as a baseline to determine whether interactions appear with "unusual" frequency among active elements (class-1 observations). Once again, these comparisons rely on human judgment to determine when results are sufficiently different. These choices depend on the domain context and how the problem has been translated. They should be transparently communicated by the researcher in the PCS documentation.
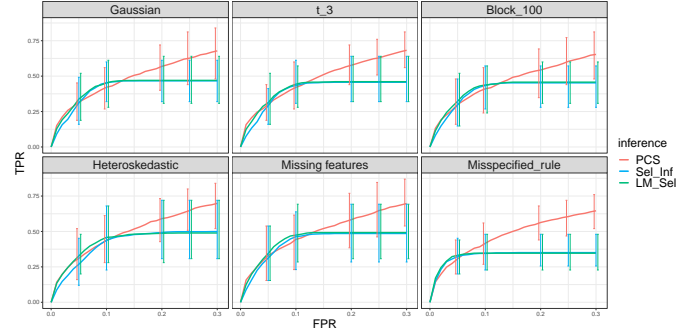

Fig. 2. ROC curves for feature selection in linear model setting with $n = 250$ observations. Each plot corresponds to a different generative model.

D. PCS inference simulation studies in sparse linear models.
We tested PCS inference in an extensive set of data-inspired simulation experiments in the sparse linear model setting that has been widely studied by the statistics community over the past two decades (SI Appendix). In total, we considered 6 distinct generative models intended to reflect some of the issues that arise in practice. We compared our proposed PCS inference procedure with selective inference and asymptotic normality results using ROC curves. These provide a useful criterion to assess false positives and true positives, which are both important considerations in settings where resources dictate how many findings can be evaluated in follow-up analyses/experiments. Across all models, PCS inference compares favorably to both selective inference and asymptotic normality results (Fig. 4). However, we note that the principal advantage of PCS inference is that it can be easily generalized to more complex settings faced by data scientists today as in the two examples described above.

4. PCS documentation

The PCS framework includes an accompanying R Markdown or Jupyter (iPython) Notebook, which seamlessly integrates narratives, codes, and analyses. These narratives are necessary to describe the domain problem and support assumptions and choices made by the data scientist regarding computational platform, data cleaning and preprocessing, data visualization, model/algorithm, prediction metric, prediction evaluation, stability target, data and algorithm/model perturbations, stability metric, and data conclusions in the context of the domain problem. These narratives should be based on referenced prior knowledge and an understanding of the data collection process, including design principles or rationales. The narratives in the PCS documentation help bridge or connect the two parallel universes of reality and models/algorithms that exist in the mathematical world (Fig. 3). In addition to narratives justifying human judgment calls (possibly with data evidence), PCS documentation should include all codes used to generate data results with links to sources of data and metadata.

We propose the following steps in a notebook[¶¶]:

1. Domain problem formulation (narrative). Clearly state the real-world question and describe prior work related to this question. Indicate how this question can be answered in the context of a model or analysis.

[§§]A null hypothesis may correspond to multiple data or model/algorithm perturbations. We focus on a single data perturbation here for simplicity.

[¶¶]This list is reminiscent of the list in the "data wisdom for data science" blog that one of the authors wrote at http://www.odbms.org/2015/04/data-wisdom-for-data-science/
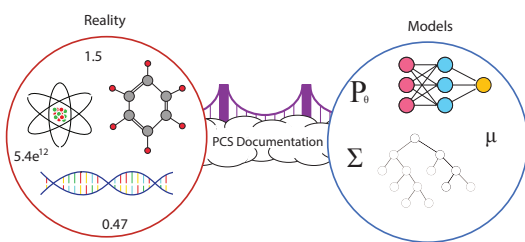
Fig. 3. Assumptions made throughout the DSLC allow researchers to use models as an approximation of reality. Narratives provided in PCS documentation can help justify assumptions to connect these two worlds.

2. Data collection and storage (narrative). Describe how the data were generated, including experimental design principles, and reasons why data is relevant to answer the domain question. Describe where data is stored and how it can be accessed by others.

3. Data cleaning and preprocessing (narrative, code, visualization). Describe steps taken to convert raw data into data used for analysis, and why these preprocessing steps are justified. Ask whether more than one preprocessing methods should be used and examine their impacts on the final data results.

4. Exploratory data analysis (narrative, code, visualization). Describe any preliminary analyses that influenced modeling decisions or conclusions along with code and visualizations to support these decisions.

5. Modeling and Post-hoc analysis (narrative, code, visualization). Carry out PCS inference in the context of the domain question. Specify appropriate model and data perturbations. If necessary, specify null hypotheses and associated perturbations.

6. Interpretation of results (narrative and visualization). Translate the data results to draw conclusions and/or make recommendations in the context of domain problem.

This documentation gives the reader as much information as possible to make informed judgments regarding the evidence and process for drawing a data conclusion in the DSLC. A case study of the PCS framework in the genomics problem discussed earlier is documented on Zenodo.

5. PCS recommendation system for scientific hypothesis generation

In general, causality implies predictability and stability over many experimental conditions; but not vice versa. The causal inference community has long acknowledged connections between stability and estimates of causal effects. For instance, many researchers have studied paradoxes surrounding associations that lead to instable estimates of causal effects (52–54). Estimates in the Neyman-Rubin potential outcomes framework rely on a stable treatment across observational units (55, 56). Sensitivity analyses test the stability of a causal effect relative to unmeasured confounding (57, 58). Stability, particularly with respect to predictions across experimental interventions,

has even been proposed as a criteria to establish certain causal relationships under the name "invariance" (39, 59–62).

PCS inference builds on these ideas, using stability and predictability to rank target estimates for further studies, including follow-up experiments. In our recent works on DeepTune (32), iterative random forests (iRF) (33), and signed iterative random forests (siRF) (43), we use PCS inference to make recommendations as inputs to downstream human decisions. For example, PCS inference suggested potential relationships between neurons in the visual cortex and visual stimuli as well as 3rd and 4th order interactions among biomolecules that are candidates for regulating gene expression. Predictability and stability do not replace physical experiments to prove or disprove causality. However, we hope computationally tractable analyses that demonstrate high predictability and stability suggest hypotheses or intervention experiments that have higher yields than otherwise. This hope is supported by the fact that 80% of the 2nd order interactions identified by iRF (33) had been verified in the literature by other researchers through physical experiments.

6. Conclusion

In this paper, we unified three principles of data science: predictability, computability and stability. Based on these principles, we proposed the PCS framework with inference procedures and documentation. The PCS framework aims to provide responsible, reliable, reproducible, and transparent results across the DSLC. Prediction serves a reality check, evaluating how well a model/algorithm reflects the natural phenomena that generated the data. Computability concerns with respect to algorithm efficiency determine the tractability of the DSLC. Stability relative to data and model perturbations was advocated in (15) as a minimum requirement for data results' reproducibility and interpretability.

We made important conceptual progress on stability by extending it to the entire DSLC, including problem formulation, data collection, data cleaning, and EDA. In addition, we developed PCS inference to evaluate the variability of data results with respect to a broad range of perturbations encountered in modern data science. We further proposed PCS hypothesis testing to draw comparisons with simple structure in the data, which ideally corresponds to an existing scientific theory. We demonstrated that PCS inference performs favorably in a feature selection problem through data-inspired sparse linear model simulation studies and in a genomics case study. To communicate the many human judgment calls in the DSLC, we proposed PCS documentation, which integrates narratives justifying judgment calls with reproducible codes and visualization. This documentation makes data-driven decisions as transparent as possible so that users of data results can determine whether they are reliable.

In summary, we have offered a new conceptual and practical framework to guide the DSLC, but many open problems remain. The PCS inference procedures need to be vetted in practice well beyond the case studies in this paper and in our previous works, especially by other researchers. Additional case studies will help unpack subjective human judgment calls in the context of specific domain problems. The knowledge gained from these studies can be shared and critiqued through transparent documentation. Based on feedback from practice, theoretical studies of PCS procedures in the modeling stage

are also called for to gain further insights under stylized models after sufficient empirical vetting. Finally, although there have been some theoretical studies on the connections between the three principles (see (63, 64) and references therein), much more work is necessary.

7. Acknowledgements

1. Murdoch WJ, Singh C, Kumbier K, Abbasi-Asl R, Yu B (2019) Interpretable machine learning: definitions, methods, and applications. *arXiv preprint arXiv:1901.04592*.
2. Stark PB, Saltelli A (2018) Cargo-cult statistics and scientific crisis. *Significance* 15(4):40–43.
3. Ioannidis JP (2005) Why most published research findings are false. *PLoS medicine* 2(8):e124.
4. Popperp KR (1959) *The Logic of Scientific Discovery*. (University Press).
5. Breiman L, , et al. (2001) Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical science* 16(3):199–231.
6. Stone M (1974) Cross-validatory choice and assessment of statistical predictions. *Journal of the royal statistical society. Series B (Methodological)* pp. 111–147.
7. Allen DM (1974) The relationship between variable selection and data agumentation and a method for prediction. *Technometrics* 16(1):125–127.
8. Turing AM (1937) On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London mathematical society* 2(1):230–265.
9. Hartmanis J, Stearns RE (1965) On the computational complexity of algorithms. *Transactions of the American Mathematical Society* 117:285–306.
10. Li M, Vitányi P (2008) *An introduction to Kolmogorov complexity and its applications. Texts in Computer Science*. (Springer, New York,) Vol. 9.
11. Kolmogorov AN (1963) On tables of random numbers. *Sankhyā: The Indian Journal of Statistics, Series A* pp. 369–376.
12. Fisher RA (1937) *The design of experiments*. (Oliver And Boyd; Edinburgh; London).
13. Donoho DL, Maleki A, Rahman IU, Shahram M, Stodden V (2009) Reproducible research in computational harmonic analysis. *Computing in Science & Engineering* 11(1).
14. Stark P (2018) Before reproducibility must come preproducibility. *Nature* 557(7707):613.
15. Yu B (2013) Stability. *Bernoulli* 19(4):1484–1500.
16. Manski CF (2013) *Public policy in an uncertain world: analysis and decisions*. (Harvard University Press).
17. Quenouille MH, , et al. (1949) Problems in plane sampling. *The Annals of Mathematical Statistics* 20(3):355–375.
18. Quenouille MH (1956) Notes on bias in estimation. *Biometrika* 43(3/4):353–360.
19. Tukey J (1958) Bias and confidence in not quite large samples. *Ann. Math. Statist.* 29:614.
20. Efron B (1992) Bootstrap methods: another look at the jackknife in *Breakthroughs in statistics*. (Springer), pp. 569–593.
21. Bolstad BM, Irizarry RA, Åstrand M, Speed TP (2003) A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics* 19(2):185–193.
22. Steegen S, Tuerlinckx F, Gelman A, Vanpaemel W (2016) Increasing transparency through a multiverse analysis. *Perspectives on Psychological Science* 11(5):702–712.
23. Dawid AP (1984) Present position and potential developments: Some personal views statistical theory the prequential approach. *Journal of the Royal Statistical Society: Series A (General)* 147(2):278–290.
24. Freedman DA (1991) Statistical models and shoe leather. *Sociological methodology* pp. 291–313.
25. Geisser S (1993) *Predictive Inference*. (CRC Press) Vol. 55.
26. Tibshirani R (1996) Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* pp. 267–288.
27. Breiman L (2001) Random forests. *Machine learning* 45(1):5–32.
28. Gneiting T, Raftery AE (2007) Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association* 102(477):359–378.
29. Wolpert L (1969) Positional information and the spatial pattern of cellular differentiation. *Journal of theoretical biology* 25(1):1–47.
30. Fushiki T (2011) Estimation of prediction error by using k-fold cross-validation. *Statistics and Computing* 21(2):137–146.
31. Robbins H, Monro S (1951) A stochastic approximation method. *The Annals of Mathematical Statistics* 22(3):400–407.
32. Abbasi-Asla R, et al. (2019) The deeptune framework for modeling and characterizing neurons in visual cortex area v4.
33. Basu S, Kumbier K, Brown JB, Yu B (2018) iterative random forests to discover predictive and stable high-order interactions. *Proceedings of the National Academy of Sciences* p. 201711236.
34. Box GE (1976) Science and statistics. *Journal of the American Statistical Association* 71(356):791–799.
35. Goodfellow I, et al. (2014) Generative adversarial nets in *Advances in neural information processing systems*. pp. 2672–2680.
36. Biegler LT, Ghattas O, Heinkenschloss M, van Bloemen Waanders B (2003) Large-scale pde-constrained optimization: an introduction in *Large-Scale PDE-Constrained Optimization*. (Springer), pp. 3–13.
37. Skene A, Shaw J, Lee T (1986) Bayesian modelling and sensitivity analysis. *The Statistician* pp. 281–288.
38. Box GE (1980) Sampling and bayes' inference in scientific modelling and robustness. *Journal of the Royal Statistical Society. Series A (General)* pp. 383–430.
39. Peters J, Bühlmann P, Meinshausen N (2016) Causal inference by using invariant prediction: identification and confidence intervals. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 78(5):947–1012.
40. Heinze-Deml C, Peters J, Meinshausen N (2018) Invariant causal prediction for nonlinear models. *Journal of Causal Inference* 6(2).
41. Meinshausen N, Bühlmann P (2010) Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 72(4):417–473.
42. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15(1):1929–1958.
43. Kumbier K, Basu S, Brown JB, Celniker S, Yu B (2018) Refining interaction search through signed iterative random forests. *arXiv preprint arXiv:1810.07287*.
44. Freund Y, Schapire RE (1997) A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences* 55(1):119–139.
45. Coker B, Rudin C, King G (2018) A theory of statistical inference for ensuring the robustness of scientific results. *arXiv preprint arXiv:1804.08646*.
46. Fisher RA (1992) Statistical methods for research workers in *Breakthroughs in statistics*. (Springer), pp. 66–70.
47. Freedman D, Lane D (1983) A nonstochastic interpretation of reported significance levels. *Journal of Business & Economic Statistics* 1(4):292–298.
48. Rubino G, Tuffin B (2009) *Rare event simulation using Monte Carlo methods*. (John Wiley & Sons).
49. Bucklew J (2013) *Introduction to rare event simulation*. (Springer Science & Business Media).
50. Schuemie MJ, Ryan PB, Hripcsak G, Madigan D, Suchard MA (2018) Improving reproducibility by using high-throughput observational studies with empirical calibration. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 376(2128):20170356.
51. Elsayed GF, Cunningham JP (2017) Structure in neural population recordings: an expected byproduct of simpler phenomena? *Nature neuroscience* 20(9):1310.
52. Fisher RA (1923) Statistical tests of agreement between observation and hypothesis. *Economica* (8):139–147.
53. Cochran WG (1938) The omission or addition of an independent variate in multiple linear regression. *Supplement to the Journal of the Royal Statistical Society* 5(2):171–176.
54. Bickel PJ, Hammel EA, O'Connell JW (1975) Sex bias in graduate admissions: Data from berkeley. *Science* 187(4175):398–404.
55. Neyman J (1923) Sur les applications de la théorie des probabilités aux experiences agricoles: Essai des principes. *Roczniki Nauk Rolniczych* 10:1–51.
56. Rubin DB (1980) Randomization analysis of experimental data: The fisher randomization test comment. *Journal of the American Statistical Association* 75(371):591–593.
57. Cornfield J, et al. (1959) Smoking and lung cancer: recent evidence and a discussion of some questions. *Journal of the National Cancer institute* 22(1):173–203.
58. Ding P, VanderWeele TJ (2016) Sensitivity analysis without assumptions. *Epidemiology (Cambridge, Mass.)* 27(3):368.
59. Haavelmo T (1944) The probability approach in econometrics. *Econometrica: Journal of the Econometric Society* pp. iii–115.
60. Cartwright N (2003) Two theorems on invariance and causality. *Philosophy of Science* 70(1):203–224.
61. Schölkopf B, et al. (2012) On causal and anticausal learning. *arXiv preprint arXiv:1206.6471*.
62. Pearl J (2009) *Causality*. (Cambridge university press).
63. Hardt M, Recht B, Singer Y (2015) Train faster, generalize better: Stability of stochastic gradient descent. *arXiv preprint arXiv:1509.01240*.
64. Chen Y, Jin C, Yu B (2018) Stability and convergence trade-off of iterative optimization algorithms. *arXiv preprint arXiv:1804.01619*.
65. MacArthur S, et al. (2009) Developmental roles of 21 drosophila transcription factors are determined by quantitative differences in binding to an overlapping set of thousands of genomic regions. *Genome biology* 10(7):R80.
66. Li Xy, et al. (2008) Transcription factors bind thousands of active and inactive regions in the drosophila blastoderm. *PLoS biology* 6(2):e27.
67. Li XY, Harrison MM, Villalta JE, Kaplan T, Eisen MB (2014) Establishment of regions of genomic activity during the drosophila maternal to zygotic transition. *Elife* 3:e03737.
68. Taylor J, Tibshirani RJ (2015) Statistical learning and selective inference. *Proceedings of the National Academy of Sciences* 112(25):7629–7634.

8. Supporting Information: Simulation studies of PCS inference in the linear model setting

In this section, we consider the proposed PCS perturbation intervals through data-inspired simulation studies. We focus on feature selection in sparse linear models to demonstrate that PCS inference provides favorable results, in terms of ROC analysis, in a setting that has been intensively investigated by the statistics community in recent years. Despite its favorable performance in this simple setting, we note that the principal advantage of PCS inference is its generalizability to new situations faced by data scientists today. That is, PCS can be applied to any algorithm or analysis where one can define appropriate perturbations. In the accompanying PCS case study, we demonstrate the ease of applying PCS inference in the problem of selecting high-order, rule-based interactions from a random forest in a high-throughput genomics problem (whose data the simulation studies below are based upon).

To evaluate feature selection in the context of linear models, we considered data for 35 genomic assays measuring the binding enrichment of 23 unique TFs along 7809 segments of the genome (65–67). That is, for an observation $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$, $i = 1 \ldots, n$, x_{ij} measured the enrichment of the jth TF at the ith segment of the genome. We augmented this data with 2nd order polynomial terms for all pairwise interactions (excluding quadratic terms x_i^2), resulting in a total of $p = 35 + \binom{35}{2} = 630$ features. For a complete description of the data, see the accompanying PCS documentation. We standardized each feature and randomly selected $s = \lfloor \sqrt{p} \rfloor = 25$ active features to generate responses

$$y = \mathbf{x}^T \beta + \epsilon \qquad [8]$$

where $\mathbf{x} \in \mathbb{R}^{7809 \times 630}$ denotes the normalized matrix of features, $\beta_j = 1$ for any active feature j and 0 otherwise, and $\epsilon \in \mathbb{R}^n$ represents mean 0 noise drawn from a variety of distributions. In total, we considered 6 distinct settings with 4 noise distributions: i.i.d. Gaussian, Students t with 3 degrees of freedom, multivariate Gaussian with block covariance structure, Gaussian with variance $\sigma_i^2 \propto \|\mathbf{x}_i\|_2^2$ and two misspecified models: i.i.d. Gaussian noise with 12 active features removed prior to fitting the model, i.i.d. Gaussian noise with responses generated as

$$y = \sum_{S_j \in \mathcal{S}} \beta_{S_j} \prod_{k \in S_j} \mathbb{1}(x_k > t_k) + \epsilon \qquad [9]$$

where $\mathcal{S}$ denotes a set of randomly sampled pairs of active features.

A. Simple PCS perturbation intervals.
We evaluated selected features using the PCS perturbation intervals. Below we outline each step for constructing such intervals in the context of linear model feature selection.

1. Our prediction target was the simulated responses y and our stability target $\mathcal{T} \subseteq \{1, \ldots, p\}$ the features selected by lasso when regressing y on $\mathbf{x}$. To evaluate prediction accuracy, we randomly sampled 50% of observations as a held-out test set. Our model/algorithm perturbatiosn were given by the default values of lasso penalty parameter in the R package `glmnet` and $B = 100$ bootstrap replicates respectively.

2. We formed a set of filtered models Λ^* by taking λ corresponding to the 10 most accurate models in terms of ℓ_2 prediction error (Fig. 4). Pre-specified prediction thresholds achieved qualitatively similar results and are reported in Fig. 5. Since the goal of our analysis was feature selection, we evaluated prediction accuracy on the held-out test data. We repeated the steps below on each half of the data and averaged the final results.

3. For each $\lambda \in \Lambda^*$ and $b = 1, \ldots, 100$ we let $\mathcal{T}(\mathbf{x}^{(b)}, \lambda)$ denote the features selected for bootstrap sample b with penalty parameter λ.

4. The distribution of $\mathcal{T}$ across data and model perturbations can be summarized into a range of stability intervals. Since our goal was to compare PCS with classical statistical inference, which produces a single p-value for each feature, we computed a single stability score for each feature $j = 1 \ldots, p$:

$$sta(j) = \frac{1}{B \cdot |\Lambda^*|} \sum_{b=1}^{100} \sum_{\lambda \in \Lambda^*} \mathbb{1}(j \in \mathcal{T}(\mathbf{x}^{(b)}, \lambda))$$

Intuitively, stability scores reflect our degree of belief that a given feature is active in the model, with higher scores implying a higher degree of certainty. In practice, these scores could be used to rank features and identify the most reliable collection for further consideration (e.g. experimental validation). We note that the stability selection proposed in (41) is similar, but without the prediction error screening.

B. Results.
We compared the above PCS stability scores with asymptotic normality results applied to features selected by lasso and selective inference (68). We note that asymptotic normality and selective inference both produce p-values for each feature, while PCS produces stability scores.

Figs. 4 and 5 show ROC curves for feature selection averaged across 100 replicates of the above experiments. The ROC curve is a useful evaluation criterion to assess both false positive and true positive rates when experimental resources dictate how many selected features can be evaluated in further studies. In particular, ROC curves provide a balanced evaluation of each method's ability to identify active features while limiting false discoveries. Across all settings, PCS compares favorably to the other methods. The difference is particularly pronounced in settings where other methods fail to recover a large portion of active features ($n < p$, heteroskedastic, and misspecified model). In such settings, stability analyses allow PCS to recover more active features while still distinguishing them from inactive features. While its performance in this setting is promising, the principal advantage of PCS is its conceptual simplicity and generalizability. That is, the PCS perturbation intervals described above can be applied in any setting where data or model/algorithm perturbations can be defined, as illustrated in the genomics case study in the accompanying PCS documentation. Traditional inference procedures cannot typically handle multiple models easily.
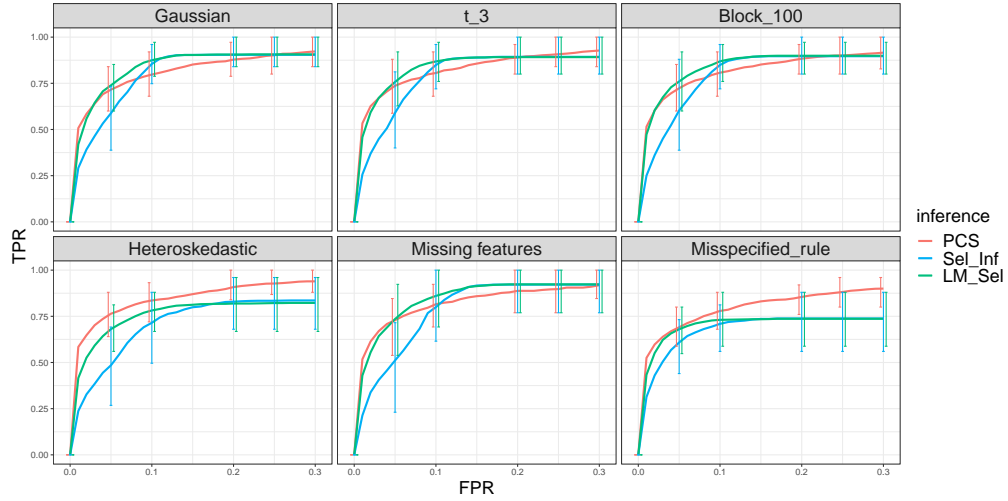
Fig. 4. ROC curves for feature selection in linear model setting, using 10 most accurate models, with $n = 1000$ observations. Each plot corresponds to a different generative model. Prediction accuracy screening for PCS inference was conducted using a pre-specified threshold.
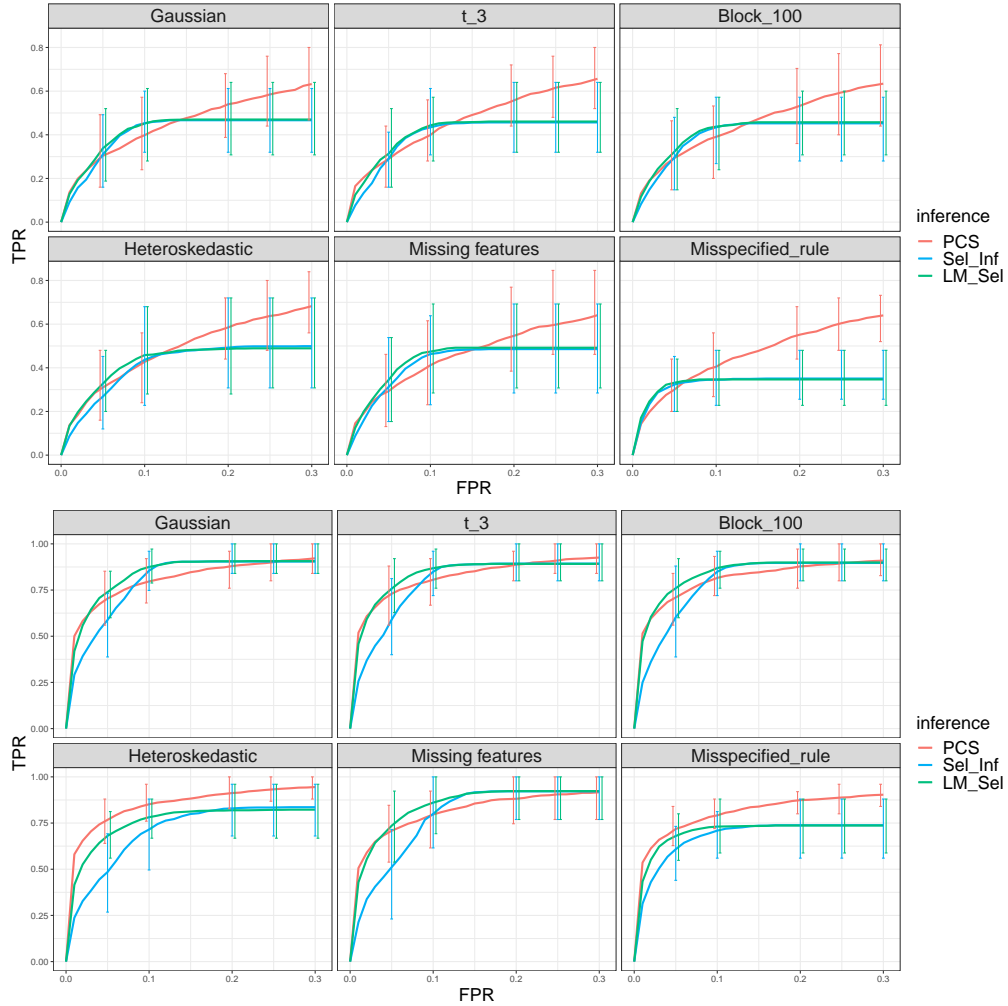

Fig. 5. ROC curves for feature selection in linear model setting, using pre-spcified threshold, with $n = 250$ (top two rows) and $n = 1000$ (bottom two rows) observations. Each plot corresponds to a different generative model. Prediction accuracy screening for PCS inference was conducted using a pre-specified threshold.