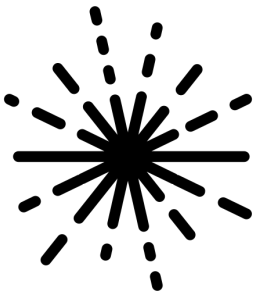

The OpenDP White Paper*

The OpenDP Team†

May 11, 2020

Abstract

OpenDP is a community effort to build a trustworthy suite of open-source tools for enabling privacy-protective analysis of sensitive personal data, focused on a library of algorithms for generating differentially private statistical releases. The target use cases for OpenDP are to enable government, industry, and academic institutions to safely and confidently share sensitive data to support scientifically oriented research and exploration in the public interest. We aim for OpenDP to flexibly grow with the rapidly advancing science of differential privacy, and be a pathway to bring the newest algorithmic developments to a wide array of practitioners.

OpenDP is led by Faculty Directors Gary King and Salil Vadhan and an Executive Committee at Harvard University, funded in part by a grant from the Sloan Foundation. Its efforts so far have included implementing a differentially private curator application in collaboration with Microsoft, and developing a framework for a community-driven OpenDP Commons through the work of an Ad Hoc Design Committee including external experts. Going forward, the project plans to engage with a wide community of stakeholders, establish partnerships with a wide variety of groups from industry, academia, and government, and adopt increasing levels of community governance.

†Contributors to this white paper include the OpenDP Executive committee (Mercè Crosas, James Honaker, Gary King, Mike Phelan, Salil Vadhan, and Annie Wu), the OpenDP Design Committee (Mercè Crosas, Marco Gaboardi, Michael Hay, James Honaker, Gary King, Aleksandra Korolova, Ilya Mironov, and Salil Vadhan), Christopher Bavitz (WilmerHale Clinical Professor of Law, Harvard Law School and Managing Director, Cyberlaw Clinic at the Berkman Klein Center for Internet and Society) and Joshua Allen (Principal Data Scientist, Microsoft). See Section 3.2 for titles and affiliations of the committee members.

Contents

1 Overview

Traditional approaches for sharing sensitive data are now well known to be ineffective in protecting privacy. Differential Privacy [14], deriving from roots in cryptography, is a strong mathematical criterion for privacy preservation that also allows for rich statistical analysis of sensitive data. Differentially private algorithms are designed by carefully introducing random noise, censoring, and other procedures designed to obscure the effect of any individual data subject on any statistical release.

Despite fast growing demand from academia, government agencies, human-subjects research communities, industry, and data archivists, practical adoption of differential privacy remains slow. To address this problem, we propose to create *OpenDP*, with four parts: (1) a suite of open-source differentially private software tools that are validated and trustworthy; (2) a community that comes together to define "validated" and "trustworthy", guides intellectual and organizational decisions, and seeks out new applications and partners; (3) a set of institutions that support the software and community efforts; and (4) new intellectual directions for the field that emerge from our collective effort to bring differential privacy theory to a wide variety of applications and to use novel applications to motivate new theoretical ideas.

Some of the use cases we envision for OpenDP are to enable:

- Archival data repositories, such as Dataverse, ICPSR, and Dryad, to offer academic researchers privacy-preserving access to sensitive data. This would allow both novel secondary reuse and replication access to data that otherwise is commonly locked away in archives, or shared using ineffective methods like "deidentification" that fail to protect privacy.

- Government agencies to safely share sensitive data with researchers, data-driven policy makers, and the broader public. For example, although the US Census has invested heavily in differential privacy, they continue to lack a general-purpose library of algorithms they can promote to other Federal statistical agencies who seek to draw on their expertise.

- Companies to share data on their users and customers with academic researchers (as in the Social Science One project, in which Facebook made differentially private data available for researchers to study the effects of social media on democracy and elections) or with institutions that bring together several such datasets. This enables groundbreaking research on novel informative datasets, often of important social consequence, without the expense of one-off researcher solutions or the risk of violating user trust (as in the Cambridge Analytica incident).

- Collaborations between government, industry, and academia to provide greater access to and analysis of data that aids in understanding and combating the spread of disease, such as the current COVID-19 pandemic and any future waves of it.[1]

- Additionally, beyond the context of sensitive data, Differential Privacy has been shown to guard against overfitting and to ensure generalization, so OpenDP tools could also be used for increasing the robustness of empirical findings [12, 11, 29, 13].

[1] See Google's recent sharing of differentially private COVID-19 mobility data (https://www.blog.google/technology/health/covid-19-community-mobility-reports) and the article "Aggregated mobility data could help fight COVID-19." Science (New York, NY) (March 23, 2020).

Differential privacy has started to have large-scale deployments in industry and government (Google, Apple, Microsoft, Uber, US Census Bureau) and there is also an extensive body of implementation and experimental work in academia. Most of this work falls short of our goals for OpenDP as they:

1. are not designed to facilitate contributions and vetting by the differential privacy community at large,
2. are highly tailored to a specific application, with particular data or analyses to be supported,
3. and/or require more expertise in computer science or DP than our anticipated users would have.

We envision OpenDP as a larger open-source project for the differential privacy community to develop general-purpose, usable, and scalable tools for differential privacy (DP) in which users can have confidence. Existing implementations of differential privacy in academia and industry, including the system that the OpenDP team is building in collaboration with Microsoft, can serve as starting points for components of OpenDP, which we envision will continue to grow and evolve through the community that builds around it.

Although we see many more potential use cases for researchers, industry, policymakers and the broader public, we are focused on opening otherwise siloed and sequestered sensitive data to support scientifically oriented research and exploration in the public interest.

The overall structure and governance of the OpenDP software will be divided as follows and as illustrated in Figure 1.

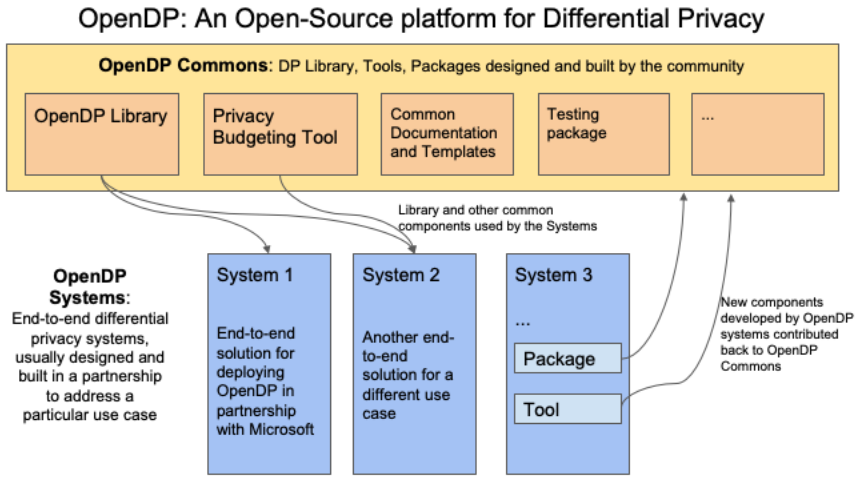

Figure 1: *End-to-end privacy preserving systems, utilizing the OpenDP Commons components.*

- *OpenDP Commons:* This community-driven layer of OpenDP includes a common differential privacy library and common tools and packages that can be used to build end-to-end differentially private systems. The governance for this layer facilitates contributions and vetting by the community, as well as reviews, guidance, and guarantees for using the library and tools.

- *OpenDP Systems:* This layer of OpenDP builds on top of the common layer. It will use the OpenDP common library and optionally other OpenDP tools or packages to develop end-to-end systems that address one or more use cases relevant to research, government, and/or industry. These systems can be thought of as "products" ready to be deployed for practical use, and will typically be built in collaboration or partnership with other organizations to provide a solution for a specific need.

- *Interaction between OpenDP common components and systems:* OpenDP systems by definition will be using the OpenDP library (once it exists) and possibly other common components. Additional code and technologies will be developed and integrated to create the end-to-end system. Whenever suitable, new components developed by the systems will be contributed back to the OpenDP common layer as open-source tools or packages reusable by the community.

As in Figure 1, we envision that multiple end-to-end systems for preserving privacy will be constructed through the OpenDP community, utilizing shared components from the OpenDP Commons. The first such systemis a collaborative project joint with Microsoft, building a curator for releasing differentially private statistical releases in a cloud environment. It has a core library of DP components that can be knitted together by an computational graph. The first release of this system is anticipated in the Spring of 2020, after which Microsoft will start using it internally for its own sensitive analytics, as well as with client partners. By the end of 2020, we expect that this system, integrated with the OpenDP Commons library and a graphical user interface modelled on PSI [18], will be a "Minimum Viable Product (MVP)" ready for adoption in OpenDP use cases, including beta deployment in Dataverse 56 data repositories. In addition, the OpenDP Commons will have the components needed for other partners to develop additional end-to-end OpenDP Systems. We expect other systems to follow the same process, and build additional end-to-end systems, while adding to the OpenDP Commons. The specification of the components to be included in the MVP will be finalized based on feedback at the Community Meeting.

Given this overall structure, our vision for OpenDP is comprised of the following principles:

1. *Use Cases:* Our goal is to open otherwise siloed and sequestered sensitive data to support scientifically oriented research and exploration in the public interest, including data shared from companies, government agencies, and research data repositories.

2. *Governance:* The goals of the OpenDP governance model including ensuring that the software is trustworthy (in particular, has correct and secure implementations of differential privacy), that the software continues to develop rapidly and serve the overall mission, and that all important stakeholders are heard.

3. *Programming Framework:* The programming framework for the core library of differentially private algorithms in the OpenDP Commons should enable the library to expand with the rapidly advancing differential privacy literature, while also ensuring the trustworthiness of code contributions and enabling the design of efficient, high-utility OpenDP Systems.

4. *Statistical Functionality:* The OpenDP software must provide statistical functionality that is useful for the researchers who will analyze data through it. In particular, it is crucial that the library of differentially private algorithms exposes measures of utility and uncertainty

that will help researchers avoid drawing incorrect conclusions due to the noise introduced for privacy.

5. *Collaborations:* OpenDP must develop fruitful collaborations with other groups in academia, industry, and government that can contribute to software development, apply OpenDP tools to important use cases, and provide personnel and financial resources to support OpenDP.

6. *System Integrations:* The OpenDP software must integrate seamlessly with the storage and compute infrastructure arising in use cases in order for the deployed systems to provide secure, efficient, and usable realizations of differential privacy.

7. *Community:* OpenDP must build a diverse, inclusive, and vibrant community, who will be motivated to carry its mission forward far into the future.

Each of these principles is described in more detail in the following sections (which can mostly be read independently based on interest). Following these principles, we aim for OpenDP to become a standard body of trusted and open-source implementations of differentially private algorithms for statistical analysis on sensitive data, and a pathway that rapidly brings the newest algorithmic developments from the theoretical literatures to a wide array of practitioners.

As such we want to build a long-lasting open-source community. To accomplish this we plan to use this workshop to kick off a two-phase deployment. The groundwork leading into this we have considered "Phase Zero". A timeline is shown in Figure 2 and with a goal of a deployable DP solution and a coalesced community of privacy research experts and practitioners within the next year.

	Timeline	**Key Deliverables**	
Phase Zero	0-6 months	Community Building, Organizational Workshops, Architectural Specifications, Board and Leadership Roles Filled.	2019 Q3, Q4
Phase One	7-12 months	Budgeting Graphical User Wireframe, Large Scale Data Engine, Prototype Library Released (from joint Microsoft project), Security Engineering.	2020 Q1, Q2 (Now)
	13-18 months	Budgeting Graphical User Completed, OpenDP Library and Programming Framework, Continued Security Engineering, Minimum Viable Product (MVP) Released.	2020 Q3, Q4
Phase Two	18-24 months	Expanded Use Case Partner Development, Additional OpenDP Library Contributions, Architecture and Deployment Improvements.	2021 Q1, Q2
	24-30 months	New Use Case Outreach Continued Advanced Library Contributions Form Steering Committee	2021 Q3, Q4

Figure 2: Timeline

2 Use Cases

As discussed in Section 1, the highest-priority use cases categories we envision for OpenDP are to enable:

- Archival data repositories to offer academic researchers privacy-preserving access to sensitive data.
- Government agencies to safely share sensitive data with researchers, data-driven policy makers, and the broader public.
- Companies to share data on their users and customers with academic researchers or with institutions that bring together several such datasets.
- Collaborations between government, industry, and academia to provide greater access to and analysis of data that aids in understanding and combating the spread of disease.
- Using the connections between Differential Privacy and generalization as a tool to prevent overfitting and increase the robustness of empirical findings.

These use cases lead us to an initial focus on the "centralized model" of differential privacy, in which there is a trusted "curator" that can store and compute[2] on the entire sensitive dataset to produce statistical releases or synthetic data (which are infused with noise in order to satisfy differential privacy). (An alternative model for differential privacy is the "local model" where noise infusion is done by the data subjects themselves before sending anything to the (now untrusted) curator; this has been used in commercial deployments by companies such as Google, Apple, and Microsoft for collecting data from their customers. An intermediate model between the centralized and local models is the "multiparty model" where several curators hold sensitive datasets about different subjects (e.g., hospitals, each with data about their patients) and the goal is to allow for statistical analysis over the union of the datasets. However, the reduced need for trust in these models has a significant price, either in the accuracy of statistics computed or in computational complexity coming from use of cryptographic techniques like secure multiparty computation. These models would require a significantly different and more complex software architecture than our proposed initial suite of tools for centralized differential privacy.

Also because of the above use cases, we focus on statistical analysis aimed at inferring valid research conclusions rather than large-scale machine learning, which may require software that is specifically architected to execute at high efficiency on specialized hardware (e.g. Google's TPUs).

Nevertheless the Programming Framework proposed for the library in the OpenDP Commons (see Section 4) seems to be still applicable to the local, multiparty, and machine learning scenarios, and in the future OpenDP could be expanded to support these.

We are very open to the use cases that are prioritized for OpenDP to be driven by community interest and researcher capabilities, but we now give some examples of the above use cases above as a way to highlighting the role of OpenDP to achieve societal and scientific impact.

[2]For more discusssion on curator models see section 2.4.

2.1 Example of Data repositories use cases: Dataverse

Dataverse [25, 7, 8, 26], developed at Harvard's Institute for Quantitative Social Science (IQSS) since 2006, enables researchers to share their datasets with the research community through an easy-to-use, customizable web interface, keeping control of and gaining credit for their data while the underlying infrastructure provides robust support for good data archival and management practices. The Dataverse software has been installed and serves as a research data repository in more than 50 institutions worldwide. The Dataverse repository hosted at Harvard University[3] is open to all researchers, and contains one of the largest collections of small to medium-sized research data in the world.

At its heart, Dataverse is open source software that automates the job of an archivist. It enables individual scholars, research groups, scholarly journals, universities, companies, and others to create virtual archives (called dataverses) that are apparently on their websites (branded as theirs, with a vast array of archival and data analytic services, but without any need for installation). Dozens of installations of the dataverse software around the world backed by major institutions, serve out thousands of these virtual archives, which offer hundreds of thousands of datasets, to millions of users. Dataverse gives researchers the ability to store the academic paper and software code uniquely associated with their research publications, and integrates with data storage services.

Dataverse repositories (like most general-purpose data repositories) currently have little support for hosting privacy-sensitive data. Datasets with sensitive information about human subjects were supposed to be "de-identified" before deposit. Unfortunately, research in data privacy starting with [31] has demonstrated convincingly that traditional de-identification does not provide privacy protection. The risks posed by re-identification are magnified as social science research becomes more data-driven, more external sources of information that can be exploited for re-identification become available, and research datasets are made accessible to the general public based on well-intentioned open data policies.

The current alternative to open data sharing in repositories like Dataverse is that researchers depositing a dataset ("data depositors") declare their dataset "restricted," in which case the dataset would not be made available for download, and the only way for other researchers to obtain access would be through contacting the data depositor and negotiating terms on an *ad hoc* basis. This approach is also unsatisfactory, as it can require the continued involvement of the data depositor, the negotiations can often take months, and thus it impedes the ability of the research community to verify, replicate, and extend work done by others.

OpenDP will aim to support three uses cases relevant to data repositories:

1. Enable variable search and exploration of sensitive datasets deposited in the repository.
2. Facilitate reproducibility of research with sensitive datasets.
3. Enable statistical analysis of sensitive datasets accessible through the repository

These three use cases are described in more detail below for Dataverse, but could also be adapted to work for other data repositories:

Enable variable search and exploration of sensitive datasets deposited in the repository: Dataverse automatically calculates variable summary statistics (counts, min/max, means,

[3] http://dataverse.harvard.edu

etc) upon the ingest of a tabular file. These summary statistics for each variable can be viewed using the Data Explorer tool, even without downloading or accessing the data file. As we integrate OpenDP with Dataverse, a data depositor should be able to generate differentially private (DP) summary statistics metadata file using an OpenDP user interface. To do this, the data depositor would select "Generate DP Summary Statistics" after the tabular data file is ingested in Dataverse, which would open the OpenDP interface. Then, she would select the privacy-loss parameter for her data file, and OpenDP would create the DP summary statistics file and Dataverse would store the newly created metadata file associated with the sensitive tabular data file. Once the dataset is published, an end-user would be able to view the summary statistics of the sensitive data file using the Data Explorer tool and without ever accessing or downloading the actual data file.

Facilitate reproducibility of research with the sensitive dataset: At least a third of the datasets deposited in Dataverse are replication data and code associated with a published scholarly paper. With OpenDP, data depositors or owners could create a differentially private release on a sensitive dataset which could be used to computationally reproduce the results of the published paper while protecting the privacy of the original dataset. In this case, like in Use Case 1 above, a data depositor would select a privacy-loss parameter through the OpenDP user interface, and use the DP statistical query interface to select and run the statistics of choice to create the appropriate replication release. The DP replication release file would be made available in the dataset and end-users would be able to download it, while the original sensitive dataset would be protected and not accessible by end-users.

Enable statistical analysis of sensitive datasets accessible through the repository: For additional flexibility, the data depositor of a sensitive dataset could allow for any researcher (end-user) to be able to run any statistic available through the OpenDP interface. In this case, the data depositor would configure the allocation of privacy-loss budgets through the OpenDP interface before releasing the dataset. Once the dataset is published, an end-user would be able to click explore for the sensitive data file, and the OpendDP statistical query interface would open. The user would not have access to the original sensitive data file but would be able to run the statistics of his choice. But only up to the point that his privacy-loss budget allows.

For Use Cases 1 and 2 to work, the data depositor should have a good understanding of what privacy-loss is, and the risks involved, to make meaningful and responsible choices when selecting the privacy-loss parameter, or, for Use Case 2, when selecting the right statistic for the DP replication data release. For Use Case 3, in addition, the end-user should understand his choices: how the number and type of queries he can run will be limited. This will need to be achieved with a clear user interface but also with documentation or possibly training to use the OpenDP tool.

The tools and interfaces employ for these use cases will be part of an end-to-end OpenDP system that integrates with data repositories, and we envision them being modelled on the prototype differentially private system PSI [18], which was specifically designed to allow non-expert data depositors and end users make effective use of differential privacy.

2.2 Example of Companies Sharing Data: Social Science One

Social Science One implements a new type of partnership between academic researchers and private industry to advance the goals of social science in understanding and solving society's greatest challenges. This partnership enables academics to analyze the increasingly rich troves of information amassed by private industry in responsible and socially beneficial ways while maintaining incentive

compatibility for all parties.

Social Science One's inaugural project on "the effects of social media on democracy and elections" offers researchers data access from Facebook in the form of a differentially private table. The differential privacy applied to the Facebook data allows researchers access to an unprecedented dataset (all URLs shared on Facebook from 2017 to June 2019) while satisfying Facebook's concerns about user privacy. However, the differentially private table contains enough noise that some academics have raised concerns about their ability to conduct statistical inference. This trade-off remains an area of active research for Social Science One.

2.3 Government Statistics

In recent years, federal, state, and local agencies have been making increasing amounts of data publicly available pursuant to open data initiatives. Government open data, from segregation and urban mobility to records of bicycle collisions, from geolocated crime incidents to geolocated pothole reports, can support a wide range of social science research about the fine-grained structure of social and political life. Notable open city portals include Boston's BARI[4] and New York's CUSP[5], and the importance of such data is highlighted by the fact that the meta-question of which governments have more open data and why has itself become a social science literature [32, 33, 37]

However, most government agencies are ill-equipped to appropriately address the privacy concerns in their data. Consequently many datasets are either withheld or released with inadequate protections. Data published to municipal open data portals often undergo an *ad hoc* de-identification process in which columns deemed identifying are removed or coarsened prior to release in microdata formats, and identifiable and sensitive information can frequently be found in the published data. For example, the open data portal for the City of Boston has published records of 311 requests containing individual-level information, such as addresses of residents requesting assistance from a program offering inspections to households with children suffering from asthma [2]. The Harvard Privacy Tools team, through ongoing engagement with open government data communities, has developed a framework by which government agencies can match modern privacy tools to the risks and intended uses of their data [2]. This framework has been advocated by a draft NIST Publication [19] providing guidance to government agencies on applying de-identification techniques. However, despite calling for use of "modern privacy tools" such as differential privacy across government, these tools are not currently available in software that can be used by non-experts, in particular by government open data managers. We hope for OpenDP to start to remedy this situation.

Government agencies will be able to use OpenDP to produce rich statistical summaries of sensitive datasets that can be shared widely without worry that the combination of released statistics will reveal individual-level information (in contrast to data de-identified using traditional means, which have repeatedly been shown to be vulnerable to re-identification). This is similar to how the US Census Bureau plans to use differential privacy to produce public-use microdata samples for the 2020 Decennial Census. In addition to tables of statistics that would be of common interest, in principle it is possible to generate differentially private "synthetic data" that reflects many statistical properties of the original dataset and thus can be treated as a safe-to-release proxy for the original dataset. (For example, this can be done by estimating the parameters of a statistical

[4] https://www.northeastern.edu/csshresearch/bostonarearesearchinitiative/boston-data-portal/
[5] https://cusp.nyu.edu

model in a differentially private way, and then generating new data points using the model with estimated parameters.)

Agencies could also provide approved researchers with the OpenDP query interface to run differentially private analyses of interest to them on the data. The reason to limit such access to approved researchers is that every query made increases the privacy loss (ε) measured by differential privacy, and thus there is a finite "privacy budget" of queries that can be made while maintaining a desired level of privacy protection. The privacy budget could be quickly exhausted with a public query interface. On their own, differential privacy tools may not provide sufficient accuracy or support for the statistical methods that a researcher needs to use to obtain publishable results. In such a case, a differentially private query interface can be used for exploratory data analysis, for example for formulating hypotheses. The final analysis could then be carried out in a more controlled manner, for example in an enclave similar to Census Research Data Centers or by having an agency statistician vet and run the analysis for the researcher.

2.4 Use Cases by Privacy Architecture: Curator Models

These use cases described for opening siloed and sequestered data presently mean we would focus on the "centralized model" of differential privacy, in which there is a trusted "curator" that can store and compute on the entire sensitive dataset to produce statistical releases or synthetic data (which are infused with noise in order to satisfy differential privacy).

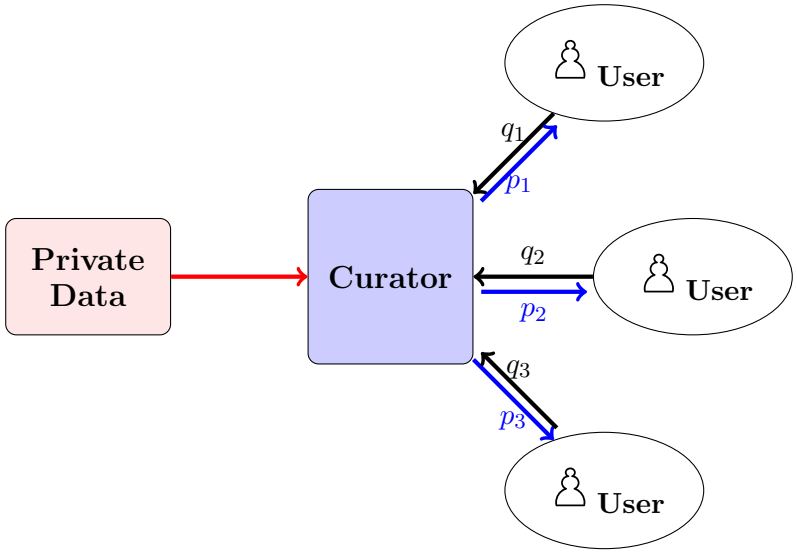

Figure 3: *The curator architecture for data privacy.*

In the curator model, private data exists in a secure location, and individuals who have been given permission can ask queries of the system. However, they can not see or interact with the raw data itself, rather they pass their queries to a curator, as in Figure 3. The curator can run the query on the private data, but instead of returning exact answers, adds sufficient noise to mask the contribution to the answer of any individual, to the degree required to preserve privacy. The user

can then do anything they want with that provided answer, without any further threat to privacy. It is key to understand that differential privacy is not an algorithm, but a definition. If an algorithm can be mathematically proven to meet that definition then it inherits all the privacy preserving properties of differential privacy. One key property is that any combination of differentially private algorithms is itself differentially private, allowing an analyst to build up complex analyses, workflows or programs. Moreover, we can formally reason about the total worst-case privacy leakage from any combination of releases, and cut off access to the database when we reach a comfortable limit of information leakage, before reidentification or other attacks can take place.

2.5 Use Cases by Analysis Type

These use case descriptions so far have focused on domains and applications. Another way to approach deciding the initial use case focus of OpenDP is to target particular methods of analysis. We list some data science and statistics *verticals* in the table below. There are not bright lines dividing these categories, and features in the library relevant to one will often be relevant to several. However, understanding what the typical scale of data, analysis methods, and most common software stacks to integrate with in each of these verticals does help aid a discussion of what to initially prioritize in the support of the OpenDP Library and other Commons components, as well as what architecture they will need. Our presently described usecases cluster under a statistical research and exploratory data analysis (EDA) grouping, which leads us currently to prioritize methods and infrastructure for these purposes, but community interest in other use cases with other needs would reorder priorities.

	Machine Learning	Data Mining	EDA and Statistics	Financial Reporting	Online Analytical Processing
Examples	Social Science One use case		Dataverse, Gov data, Social Science One use cases	Sales Tracking, Marketing	Quarterly Financial, Monthly Inventory
Update Frequency	One time (or small number)	As needed	One time (or small number)	Quarterly, Monthly	Daily, Hourly
Scale	Big	Big	Smallish	Medium	Medium
Query Complexity	Varies, typically high	Varies, typically moderate	Varies	Static, predetermined workload	Simple Count, Sum, Mean with simple predicates, joins
Data Structure	Wide distribution, often denormalized	Wide distribution, often normalized	Informal structure, "tidy"	Highly Denormalized, "Data Warehouse"	Denormalized with intermediate aggregation
Privacy Semantics	Privacy at multiple resolutions	Privacy at multiple resolutions	Person is row	Person or Group is row	Privacy at multiple resolutions

3 Governance

3.1 Goals

The goals for the OpenDP governance structure are to:

- **Enable and encourage constant contribution of code from a world-wide diverse community.** OpenDP should benefit from the rapidly growing state of knowledge on both the underlying science of DP as well as its practical deployment.
- **Enable and encourage adoption of OpenDP software for the target use cases.** One of the key missions of OpenDP is to open up otherwise siloed and sequestered sensitive data to support scientifically oriented research and exploration in the public interest.
- **Facilitate collaboration and partnership with different entities, including individuals, businesses, academic institutions, government agencies, and other software projects.** These partners will be crucial for the two items above.
- **Create a diverse and inclusive OpenDP community that gives all stakeholders a voice in the future directions of the project.** OpenDP will serve a world-wide community of researchers, government agencies, companies, and data subjects by providing mathematically proven software tools to protect privacy while unlocking sensitive data for societally beneficial analyses. Having a diverse and inclusive community reflects this mission and ensures it is accomplished from the ground up.
- **Maintain a well-defined process on contribution, vetting and testing to ensure transparency and quality.** Indeed, it is important for the OpenDP software to be trustworthy so that users will be comfortable with deploying it with highly sensitive data.
- **Ensure timely and sufficient resources for meeting strategic directions and sustaining OpenDP as it grows, while allowing the community to drive its growth.** OpenDP is being incubated at Harvard, with resources coming from an initial grant from the Sloan Foundation and the ability to utilize many university services at no cost, such legal counsel, grant management, technology advisory and operational support. But even for its near-term goals, support from the university, foundations and government will to be supplemented by contributions from the private sector. In enlisting support from large players in industry, OpenDP must also keep its mission as a community driven project and maintain the balance of participation among different entities.
- **Transform and evolve through lifecycle stages of the organization.** Start-up entities have a different set of needs than mature ones. OpenDP is no exception. Its governance model needs to be flexible and responsive in order to facilitate and strengthen its growth by staying aligned with the community composition over time and the new strategic directions to embark on.

3.2 OpenDP Governance Structure

Entering the Community Meeting in May 2020, the following parts of our governance structure will already be in place:

- **Faculty Directors:** Oversee project and related committees, setting strategic direction and scientific goals for OpenDP in its current stage.
  - Gary King, Albert J. Weatherhead III University Professor, Director, IQSS, Harvard University
  - Salil Vadhan, Vicky Joseph Professor of Computer Science and Applied Mathematics, Harvard College Professor, Harvard University
- **Executive Committee:** Responsible for day-to-day operations of the OpenDP project, including resource and task management, budgeting and communication, etc.
  - Mercé Crosas, University Research Data Management Officer, HUIT, Chief Data Science and Technology Officer, IQSS, Harvard University
  - James Honaker, Research Associate, Privacy Tools Project, Harvard John A. Paulson School of Engineering and Applied Science
  - Gary King, Albert J. Weatherhead III University Professor, Director, IQSS, Harvard University
  - Michael Phelan, Senior Application Architect for OpenDP, IQSS, Harvard University
  - Salil Vadhan, Vicky Joseph Professor of Computer Science and Applied Mathematics, Harvard College Professor, Harvard University
  - Annie Wu, Program Director for the Privacy Tools Project and OpenDP, IQSS, Harvard University.
- **Ad Hoc Design Committee:** This committee was convened during the 2019-20 academic year to design the programming framework for the OpenDP Commons Library, and provide feedback on other design aspects, such as governance, use cases, and software architecture.
  - Mercè Crosas, University Research Data Management Officer, HUIT, Chief Data Science and Technology Officer, IQSS, Harvard University
  - Marco Gaboardi, Assistant Professor, Computer Science, Boston University
  - Michael Hay, Associate Professor, Computer Science, Colgate University
  - James Honaker, Research Associate, Privacy Tools Project, Harvard John A. Paulson School of Engineering and Applied Science
  - Gary King, Albert J. Weatherhead III University Professor, Director, IQSS, Harvard University
  - Aleksandra Korolova, WiSE Gabilan Assistant Professor of Computer Science, University of Southern California
  - Ilya Mironov Research Scientist, Facebook AI
  - Salil Vadhan, Vicky Joseph Professor of Computer Science and Applied Mathematics, Harvard College Professor, Harvard University
- **Advisory Board (in formation): Prof. Adam Smith** (Boston University) and **Prof. Gerome Miklau** (UMass Amerst) have agreed to chair the Advisory Board.

- **Membership:** Approximately 30 experienced and distinguished experts representing the interests of the many stakeholders that OpenDP will serve, including the DP research community, social science, medical informatics, policymakers, data repositories, open-source software development, government, and industry, as well as the vulnerable subjects in sensitive datasets.
- **Charge:** Review and provide guidance to the OpenDP Executive Committee on the broad goals of OpenDP and all aspects of its development, including governance, technical capabilities, use cases, collaborations, community engagement, and sustainability. It has a purely advisory role and does not have formal decision-making power.
- **Expectations:** Attend biannual meetings of the advisory board (one of which will typically be a physical meeting colocated with the annual OpenDP Community Meeting), view presentations and/or read reports from the OpenDP Executive Committee in advance of the meetings, and provide feedback during or after the meeting. Occasionally the Executive Committee may consult the Advisory Board for guidance when challenging questions arise between meetings.
- **Term length:** a rotation schedule and stable size for the Advisory Board will be determined once OpenDP settles into more of a steady state. We hope that initial members will be willing to serve for 1-2 years until we reach that point.

We have assembled expert involvement around our key tasks, leveraging an ongoing project on open-source software health,[6] and follow recommendations provided by the NumFocus project[7] to define OpenDP's community architecture and sustainability.

In the months following the Community Meeting, we will establish the following additional roles:

- **DP Application Leader(s):** One or more individuals (most likely faculty or researchers from outside Harvard), responsible for bringing potential applications of differential privacy to the attention of the OpenDP community, recommending the priority of each application, and leading the formation of partnerships and grant applications to support the expansion or adaptation of OpenDP for those applications. Will become part of the Executive Committee.

- **Editorial Board:** Led by an editor-in-chief, this group manages reviews of privacy proofs for pseudocode attached to code contributions (pull requests) to the OpenDP Commons Library. This board will grow over time in proportion to the amount of code contribution and include experts from the community who have become familiar with the OpenDP programming framework. Requires mathematical expertise in differential privacy and in the development and deployment of differentially private algorithms. They will provide guidance on the architectural choices in OpenDP that implicate differential privacy and the prioritization of methods from the differential privacy research literature to incorporate into OpenDP.

[6] Mercé Crosas leads a joint IMLS and Sloan funded project titled "A Proposed Quantitative Index to Understand and Evaluate the Health of Open Source Projects' (https://projects.iq.harvard.edu/osshealthindex)', which leverages the also Sloan funded CHAOSS project (https://chaoss.community/) and the IMLS funded "Lyrasis: It Takes a Village" (http://lyrasisnow.org/press-release-itav-guidebook/) project to define metrics to evaluate the health of academic open-source software projects

[7] https://numfocus.org/

- **Committers:** With write permissions on the OpenDP Commons code repositories, the committers accept code contribution via a published and transparent process to ensure fairness and visibility to the community. For contributions to the OpenDP library, a committer should (a) have the editorial board review the privacy proof for the pseudocode, (b) verify that the implementation faithfully implements what is described in the pseudocode, (c) verify that the code meets the OpenDP style guidelines, and (d) verify that there are no unresolved critical issues or questions related to the code contribution.
- **Ad hoc Security Review Committee:** This committee consists of computer security experts to be convened in Summer/Fall 2020 to review and do penetration testing on OpenDP software architecture and implementation for security vulnerabilities, side-channel attacks, etc.

The partition of the OpenDP software into OpenDP Commons and OpenDP Systems is shown in Figure 1 in Section 1. The OpenDP Commons will be actively governed by the community, with an editorial board and a committee of code committers serving as gatekeepers to ensure mathematical correctness and software quality for contributions to the library. OpenDP Systems will be governed separately, in collaboration with the partners involved in developing those systems, to facilitate rapid development and innovation.

In addition to roles in the governance of OpenDP, the broader community includes contributors, both from academia and industry, and use case partners. The OpenDP Program Director, together with the DP Application Leader, are charged with driving the collaborative process, and collecting input from all the stakeholders: the contributors of code to OpenDP, the academic, government, and industry organizations that will deploy and use OpenDP, and individual end-users, so as to help set priorities, grow the community, and foster adoption.

Transition to Steady State. OpenDP aims to grow from its current incubation stage to a stable community effort that will be sustained far into the future. A trusted plan for growth and stability also helps ensure contributions by the community in the present, since contributors know their efforts will be maintained and continue to be utilized. We will hold Annual or Biannual Community Meetings to foster interaction and show the health of the project.

After OpenDP completes its growth stages and reaches a mature and stable stage of operation, we expect it to transition to a governance model resembling that of one or more of the existing large open-source software projects. In particular, we propose that a Steering Committee eventually be created to represent the interests of the many stakeholders in the OpenDP community, and to drive periodic reviews and revisions of the goals and roadmap for OpenDP. It can be separated into Community and Technical Councils, each with subcommittees where needed. Key roles typical to open-source projects, such as maintainers, committers and contributors, will be represented on boards and committees, when there are a sufficiently large number of active participants on the project.

3.3 Intellectual Property

3.3.1 Open Licenses

Overview. OpenDP has identified the MIT License as its preferred open source license for inbound and outbound contributions of software code to and by the project.[8] This section of the whitepaper provides a brief description of the MIT license and assesses some of the factors that went into its selection.

One License or Multiple Licenses? It is not strictly necessary for a platform such as OpenDP to choose a single license. The platform could be constructed in a way that allows for individual code contributors to select which license applies to their code, and for code users to download code subject to the terms of each contributor's selected license. A platform like GitHub operates in this way, with each piece of code subject to the terms of a license applied by the uploader, but no need for uniformity across projects as to license selection.

The primary reasons for expressing a preference for a single license and requiring that all contributions be made subject to that license involve simplicity and interoperability. The contributions to OpenDP — unlike contributions to a more general-purpose platform — will be designed to work together. Having all code available through the OpenDP platform licensed under a single license will ensure a level of interoperability that is desirable.

The primary downside of insisting upon a single license for all code is that some contributors may not be in a position to agree to the terms of the MIT License. There also may be existing differential privacy applications available under other open source software licenses that could not become part of the OpenDP corpus if OpenDP insists on using the MIT License. These risks seem small and are worth taking, at least at this early stage in the project. If, at a later date, the project wishes to include software code available under other licenses on the platform, nothing precludes OpenDP from reassessing this decision at a later date.

The MIT License. The MIT license is among the shortest and most permissive open source licenses that is widely available. In a nutshell, the License provides that one may use the code that is the subject of the license, without restriction, subject only to two sets of conditions. First, users of the code must include a specified copyright notice identifying its owner. Second, the License has the licensor disclaiming warranties associated with the code.

Why MIT License? OpenDP's primary goal in selecting an open source software license to use with respect to code on the platform is to ensure maximum use and uptake of code available via OpenDP by the greatest possible number of users. The brevity and simplicity of the MIT license lends itself to this goal of broad and widescale distribution. The MIT License is highly compatible with other forms of open source software licenses, meaning that code on OpenDP is likely to work not just with other code on OpenDP but with other code that is more widely available.

The high degree of compatibility of the MIT License with other open source software license gives it a slight advantage over one of the other contenders, the Apache license, which can have interoperability issues (including with respect to the GPL).

[8] The text of the MIT License is available at https://opensource.org/licenses/MIT; a copy of the license terms is attached as Appendix B.

Non-Code Contributions. The OpenDP project expects that there will be some copyrightable contributions to the project (and outputs from the project) other than software code. With respect to those non-code contributions, the project plans to require use of a Creative Commons Attribution International 4.0 license (often abbreviated "CC BY 4.0").[9]

Similar in spirit to the MIT License, the CC BY 4.0 license is expansive and permits a broad variety of use and re-use of copyrighted materials subject only to the requirement of attribution. The OpenDP project believes that this license will maximize use and re-use of OpenDP content and minimize barriers to entry. As with the MIT License, there is a risk that a contributor might not be in a position to contribute non-code copyrighted materials subject to the terms of such a broad license. But, that risk seems small and is offset by the significant benefits of using such a straightforward and unrestricted license.

3.3.2 Contributor License Agreements

The project is faced with a decision whether those who work on code in the OpenDP repository should agree to the terms of Contributor License Agreements (CLAs). A sample CLA is attached hereto as Appendix C.

CLAs offer several benefits to open source software development projects such as OpenDP, including the following:

- CLAs centralize ownership of intellectual property in a single entity, which then handles the outbound licensing under a given open source license.
- CLAs help the party managing the open source development project to ensure that contributors have the necessary rights to contribute to the project.
- CLAs may include (as the draft attached as Exhibit B does) an indication from the contributor's employer (which might otherwise claim intellectual property rights in the contribution), confirming that the employer has signed off on the employee's contribution to the project. This avoids potential pitfalls involving not just contributors but contributors' employers, who might otherwise later claim rights in code available on the platform.

The primary risk associated with using CLAs is that some subset of contributors may perceive them to be complex or burdensome and may elect not to participate in the project if CLAs are required. Several things about OpenDP suggest this should not be a significant concern, and — to the extent it is a concern — OpenDP can take steps to mitigate any risk. In particular:

- Many of the early-stage contributors to OpenDP will be known to the project, and OpenDP can work with those early-stage contributors to ensure the terms of the CLA are satisfactory.
- OpenDP can make the license terms available in "legalese" and in "plain English" versions, with significant explanatory and other material to walk contributors through the process of signing the agreements.
- OpenDP can affirmatively acknowledge and represent its own plans to make contributions of code available to the world under the terms of an open source software license (in this

[9]The text of the CC BY 4.0 license is available at https://opensource.org/licenses/MIT.

case, the MIT License), which should allay contributors' concerns that OpenDP will exercise excessive control.

3.4 Diversity and Code of Conduct

A major goal at this stage of the OpenDP project is to create a vibrant, diverse, inclusive, and global community of researchers, developers, and practitioners from academia, industry, and government to actively contribute to the growth and adoption of the OpenDP platform and tools for privacy-protective data sharing in the public interest. Because such tools can have the greatest impact on minorities and vulnerable communities, it is important that their voices be heard as these tools are designed and developed. For example, the use of differential privacy for the 2020 US Census can offer greater privacy to minority communities that may fear being targeted by immigration officials, but the noise introduced will also have a larger relative effect on their counts and hence may make it difficult to detect and litigate against discriminatory gerrymandering.

As a declaration of values and a signal of who we aim to be as a community, we will have a code of conduct for our workshops that solidify our aspirations and commitments. We propose to model our code of conduct on that of NumFOCUS,[10] which has detailed standards of behavior and reporting mechanisms. For now, we adopt the following adaptation of the short version of NumFOCUS' code:

> OpenDP Code of Conduct
>
> Be kind and behave professionally. Be inclusive of people from different backgrounds and of different ways of thinking.
>
> Sexual, racial or exclusionary language or imagery is offensive to others and destructive to the entire community.
>
> We are dedicated to building a harassment-free community for everyone, regardless of gender, sexual orientation, gender identity and expression, disability, physical appearance, body size, race, or religion. We do not tolerate harassment of community members in any form.
>
> Thank you for helping make OpenDP a welcoming, friendly community for all.

Indeed, thank you for helping make OpenDP a welcoming, friendly community for all.

[10] https://numfocus.org/code-of-conduct

4 Programming Framework

By Fall 2020, we will launch the OpenDP Commons, by releasing a thoroughly vetted library of differentially private algorithms and opening its repository for external contributions. To this end, members of our Design Committee have developed a proposed programming framework for implementations of differentially private algorithms in OpenDP, incorporating and expanding on insights coming from existing differential privacy programming frameworks such as PinQ [27], Fuzz [20], and εketelo [36], as well as our own experiences from PSI [18] and our collaborative project with Microsoft.

The proposed programming framework is described in detail in a companion paper [17]. Here we include extracts from that paper to provide an overview of the framework and its implications. We encourage interested readers to look to the other paper for technical details, such as the mathematical specification of the library architecture and illustrative implementations in code snippets.

4.1 Goals

There are a number of goals we seek to achieve with the programming framework and language choice:

Extensibility We would like the OpenDP library to be able to expand and advance together with the rapidly growing differential privacy literature, through external contributions to the codebase. This leads to a number of the other desidarata listed below.

Flexibility The programming framework should be flexible enough to incorporate the vast majority of existing and future algorithmic developments in the differential privacy literature. It should also be able to support many variants of differential privacy. These variants can differ in the type of the underlying sensitive datasets and granularity of privacy (e.g. not only tabular datasets with record level-privacy, but also graph datasets with node-level privacy and datastreams with user-level privacy), as well as in the distance measure between probability distributions on adjacent datasets (e.g. not only pure and approximate differential privacy, but also Rényi and concentrated differential privacy).

Verifiability External code contributions need to be verified to actually provide the differential privacy properties they promise. We seek a programming framework that makes it easier for the OpenDP Editorial Board and OpenDP Committers to verify correctness of contributions. Ideally, most contributions will be written by combining existing components of the library with built-in composition primitives, so that the privacy properties are automatically derived. Human verification should mostly be limited to verifying mathematical proofs for new building blocks, which should usually be small components with clear specifications. At the same time, if an error is later found in a proof for a building block, it should be possible to correct that component locally and have the correction propagate throughout all the code that makes use of that building block.

Programmability The programming framework should make it relatively easy for programmers and researchers to implement their new differentially private algorithms, without having to

learn entirely new programming paradigms or having to face excessive code annotation burdens.

Modularity The library should be composed of modular and general-purpose components that can be reused in many differentially private algorithms without having to rewrite essentially the same code. This supports extensibility, verifiability, and programmability.

Usability It should be easy to use the OpenDP Library to build a wide variety of DP Systems that are useful for OpenDP's target use cases. These may have varying front-end user interfaces (e.g. programming in Python notebooks versus a graphical user interface) and varying back-end storage, compute, and security capabilities.

Efficiency It should be possible to compile algorithms implemented in the programming framework to execute efficiently in the compute and storage environments that will occur in the aforementioned systems.

Utility It is important that the algorithms in the library expose their utility or accuracy properties to users, both prior to being executed (so that "privacy loss budget" is not wasted on useless computations) and after being executed (so that analysts do not draw incorrect statistical conclusions). When possible, algorithms should expose uncertainty measures that take into account both the noise due to privacy and the statistical sampling error.

4.2 Elements of the Framework

Differentially private programming frameworks like PinQ [27], Fuzz [20], and εketelo [36] are based on two kinds of operators:

- *Measurements*, which are randomized functions that take a sensitive private dataset and produce a differentially private output. Specifically, it is required that "close" datasets (differing on one individual's data) map to "close" output distributions (where the probability of any output differs by factor of at most roughly $1 + \varepsilon$). The classic Laplace mechanism, which computes a sum and adds noise is an example.

- *Transformations*, which are functions that manipulate private datasets before measurements are applied. It is required that transformations are *stable* in the sense that "close" datasets map to "close" datasets. An example is a data transformation that filters the original dataset to focus on the individuals with certain characteristics of interest.

The companion paper [17] proposes three orthogonal extensions to the basic framework:

1. A framework for ensuring that measurements and transformations have their claimed privacy or stability properties, by only allowing procedures that have been vetted (e.g. through the OpenDP editorial board reviewing a written proof) the right to directly construct measurements and transformations.

2. Allowing for many sensitive datatypes (not just multisets), many measures of distance between sensitive data items (not symmetric difference), and many measures of closeness between probability distributions (not just the multiplicative notion used to define pure differential

privacy, but also multi-parameter notions like (ε, δ) differential privacy). With this generalization, stability and privacy are no longer measured by single parameters, but are given by *relations* that tell us whether a given kind of "closeness" of inputs implies a given kind of "closeness" of outputs.

3. Generalizing measurements to spawn interactive query procedures (randomized state machines called *queryables*). This allows for modelling adaptive composition theorems as measurements, as well as more sophisticated interactive DP algorithms. It also provides for in-library (and thus vetted) mechanisms for managing privacy budgets, rather than delegating this to external systems.

Although we described these extensions separately for sake of exposition, we propose that all three be combined in the OpenDP Programming Framework. They combine with each other in natural ways:

- Our definition of privacy for interactive measurements readily extends to other privacy notions, in particular by simply asking that the views of the adversary on two input data items (that are close according to some distance measure) are close according to any desired notion of closeness between probability distributions (capturing, for example, approximate differential privacy or concentrated differential privacy).
- The verification principle for code contributions can and should be attached to more general measurements and transformations. In particular, now a measurement or transformation is considered *valid* if its privacy or stability relation is sound: it should never claim that a false privacy or stability bound holds.

4.3 Ensuring Privacy in Implementation

Like with any other implementation of security or privacy critical code, we need to beware of places where the mathematical abstraction used in the privacy analysis may deviate from the implementation.

For example, in the framework discussed above we abstract transformations as functions T and measurements as randomized functions M. Implicit in this modelling is that when executing $T(x)$, the only private information it has access to is x, and that it does not produce any observable output other than $T(x)$. That is, we need to prevent leakage of information through *side channels*.

One known side channel of concern is the *timing channel*, where the amount of time to execute a function reveals information about its input. Standard mitigations against the timing channel include introducing delays or truncating long executions to make a function essentially constant time. This problem and side channels in general are exacerbated when a transformation or measurement is built using a user-defined function f (which may come from an interactive query issued by an adversary). For example, a stable "map" transformation applying a function f to every record in a dataset can cause privacy leakage if the function f can write to an I/O device observable to the adversary or if the function f intentionally changes its execution time based on the data record it is fed. Side-channel attacks using user-defined functions can be mitigated by some programming language support. (See Section 4.4.)

Another concern comes from the implementation of arithmetic. Like in much of statistics, in the mathematical analysis of differential privacy, we often model data using real numbers, use

continuous noise distributions, and allow probabilities that are arbitrary real numbers in $[0, 1]$. Unfortunately, as shown by Mironov [28] the discrepancy between floating-point numbers and true real numbers leads to severe violations of differential privacy. For this reason, following [4], we advocate the use of *fixed-point* numbers and integer arithmetic as much as possible in the OpenDP library. For code that is in the fully certified library, any deviations from standard arithmetic (e.g. overflow of floating-point arithmetic) should be explicitly modelled in the privacy analysis. (There may be versions of the library that are subject to less rigorous constraints for the purpose of prototyping and experimentation.)

Differentially private algorithms also rely crucially on high-quality randomness. To avoid this being a source of vulnerability, the OpenDP library should use a cryptographic-strength pseudo-random generator for its random bits.

4.4 Discussion of Implementation Language(s)

Desidarata. The choice of programming language, or languages, for the OpenDP library implementation is an important one. It has implications on performance, reliability and also impacts the degree to which the library will achieve each of the goals identified in the introduction. This section identifies some features of the programming language that might be particularly beneficial and offers a concrete proposal for OpenDP. It is likely that no single language meets all criteria and that different components of the library will be implemented using different languages. Our initial prototyping was done using Python with a mix of object-oriented and functional paradigms, as reflected in the code examples above, and many of the features identified below are a reflection of that experience.

Section 4.1 outlines high-level goals including *usability* and *programmability*. To support the goal of usability, it will be important that the library offer APIs in languages that are commonly used in OpenDP's target use cases. The most common (open-source) languages for statistical/data analysis include Python and R. By providing APIs in one or both of these languages, programmers will be able to integrate the library into the conventional data workflows. There will also be a need to integrate with a variety of back-end data access layers, which may necessitate re-implementing some core functionality in a back-end compatible language such as as SQL. It is anticipated that this re-implementation would be limited to data-intensive transformations, though prior work has also explored implementing measurements in the data layer [22]. To support the goal of programmability, it may be preferable to select a language in the more familiar imperative paradigm rather than adopt a purely functional language, though as noted below, choosing a language that has support for some functional features may be desirable.

Some specific programming language features that support the proposed framework include memory safety, encapsulation, immutability, abstract (parameterized) data types, type safety, functional features and "pure" functions, generics, and structures or object-orientation.

Proposal: Rust + Python Given the above, we propose to implement the core library using the programming language Rust. Rust is a fast-growing, imperative functional language. It is both high performance and reliable with good memory safety and type safety. It also meets many of the desiderata identified previously: structures, generics, higher-order functions, and values are immutable by default. It has a strong ownership model that gives memory safety without the necessity of garbage collection. Collectively, this makes Rust code much easier to reason about

concretely. For example, there are no dangling pointers, there are no memory races as there are in C++.[11]

Where Rust may fall short is in the programmability criteria, as it is not a widely used language (though interest in it is growing, and it has been ranked the "most loved" language on Stack Overflow, edging out Python for the last four years) and it can be challenging to program (at least in part because of the built-in safety features).

The OpenDP project has already been developing an end-to-end differential privacy system as a joint project with Microsoft, and for this project required a library to stand in while the core OpenDP library is not yet ready. The development team built a prototype library in Rust, using ideas from the PSI library [18] and some advances in the direction of those proposed in this paper. All of the developers were new to Rust, and all picked it up straightforwardly. Also, one graduate student who wished to contribute code from a new research paper was able to quickly port their code to Rust with some small direction.

Our proposal is that Rust should be the language for the robust, polished, production code that enters the maximally trusted OpenDP library. However, to encourage contributions from researchers and other developers, as well as to encourage experimentation and sandboxing of performance, the library should be able to work with Python and R code components. We propose to enable this through Python and R bindings (through a Protocol Buffer API described next). Initial code could be contributed in Python (and R), with the goal that final, vetted and performant code be ported to Rust, either by contributors when possible, or by the core OpenDP development team, or in collaboration.

APIs. Rust uses the same Foreign Function Interfaces as C++ for bindings so the bindings compiled from either Rust or C++ are identical. However, in the OpenDP project with Microsoft, the code to write these bindings has been found to be clearer than in C++. That project has created Python bindings to the prototype differential privacy library that have been heavily tested and work well. R bindings are in construction and have been found to be straightforward too.

In that project, an API to the library was built using Protocol Buffers,[12] a serialized data language, like JSON or XML, but heavily typed and very compact. This allowed for defining any sequence of calls to the library as a plan, in the style of εktelo or more specifically, a computational graph of library components. The protobuf API calls Rust components for the runtime. The Python and R bindings also call the API which calls the Rust library. In the case of Python, these bindings can be distributed on PyPI and pip installed, and executed in a Python notebook seemingly natively, although the actual execution is in Rust. The OpenDP Design Committee proposes to release a Protocol Buffer API for the core library as part of the OpenDP Commons, by which it will also offer Python and R bindings. This API would facilitate other possible bindings, as well a way to call the library in more complicated system architectures.

[11] The Microsoft Security Response Center (MSRC) released reports stating that 70% of security vulnerabilities addressed through Microsoft security patches are memory safety issues that would have been avoided in Rust,https://msrc-blog.microsoft.com/2019/07/18/we-need-a-safer-systems-programming-language/, and explicitly suggests moving away from C++ to Rust https://msrc-blog.microsoft.com/2019/07/22/why-rust-for-safe-systems-programming/.

[12] https://developers.google.com/protocol-buffers/

4.5 Using the Library

The programming framework proposed in this paper is meant to capture all the mathematical reasoning about privacy needed for use of differential privacy. However, we do not expect it to be directly used by end users; instead it is meant as a substrate to build larger, user-friendly end-to-end differential privacy systems. Here we discuss some aspects of how such a system should use the library envisioned here.

Calling the Library from a DP System.

1. The system should first determine, or elicit from its user, the following:

   (a) The sensitive dataset on which differential privacy is going to be applied.

   (b) The type of the dataset (e.g. is it a multiset of records, or social network graph, or a collection of tables). Do the records consist of a known set of attributes?

   (c) The granularity of privacy protection. For example, if we want to protect privacy at the level of individual human beings, we need to know whether every individual human corresponds to at most one record in the multiset, at most one node in a social network graph, or at most one key in a collection of tables.

   (d) The privacy measure to be used (e.g. pure, approximate, or concentrated DP), and the desired privacy-loss parameters.

   Regarding the type of the dataset, it is best for the system to make as few assumptions as possible about the data type, as long as the granularity of privacy can be well-specified, to reduce the chance that an incorrect assumption leads to a violation of privacy. For example, it is safer to assume that the dataset is a multiset of records of arbitrary type or a multiset of strings, rather than assuming it is a multiset of records that each consist of 10 floating-point numbers in the interval $[0, 1]$. Additional public, non-sensitive information about the dataset (e.g. a full schema) can typically be exploited later for greater utility without being used for the privacy calculus. Note that specifying the granularity of privacy requires keeping information in the datasets that one might be tempted to remove beforehand — in order to enforce privacy at the level of individuals in a database where an individual might contribute to multiple records, we need the data records to have a key that corresponds to distinct individuals.

2. Then the system should select an interactive measurement family from the library (typically some form of adaptive composition) that is compatible with the data type, granularity of privacy, and privacy measure, and use it to spawn a queryable on the sensitive dataset that will mediate *all* future access to the dataset.

3. If additional type information about the dataset is known (e.g. about the individual records, or even a more complex property like being a social network of bounded degree), then before applying any queries to the dataset, the library can be used to apply a stable "casting" transformation that ensures that the dataset actually has the specified properties. For example, for multisets we can apply a record-by-record transformation that maps any record that does not have the specified type to a default value with that type.

4. All queries to the dataset should be made through the initial queryable. The queryable automatically ensures that we stay within the initially specified privacy-loss budget. Deciding how to apportion the budget among current and future queries is for the system to decide, but we expect that statistical algorithms in the library will expose their utility in ways that assist a system in making such decisions (e.g. through a priori utility estimates provided analytically or through benchmarking, as discussed in Section 5).

Exposing System Features to the Library. To make efficient and safe use of the library, it is also important to expose some system features to the library:

- Exposing the data-access model to the library. What means do the library functions have to read the sensitive data? If, for example, the data is only available in a streaming format, that will limit the choice of methods in the library that can be applied.
- Exposing the capabilities of the back-end data system to the library. Global data transformations, aggregates, sorting, and joins can be performed much more efficiently if they are done by an optimized database engine than if the data needs to be extracted from the database and computed on externally. Thus, there should be a way of identifying which transformations in the library can be done by the database system. On the other hand, this means that the resulting privacy properties also depend on the database engine faithfully and securely implementing the specified transformation, without observable side effects (including timing) that might leak sensitive data. More complex capabilities that the database may have include randomized procedures like subsampling.
- Defining the partition between secure and insecure storage and compute according to the threat model (to what might the privacy adversary have access), so that private data does not cross the "privacy barrier" until after the protections of differential privacy have been applied. In addition to the original sensitive dataset, the results of transformations, and the state of queryables need to be kept secret from the privacy adversary.

User interfaces. A system should also provide a user-friendly way for data custodians and data analysts without expertise in differential privacy to make effective use of the library. In particular, we envision that many existing DP systems can be built as layers on top of the library, using the certified code in the library for all the privacy analysis. Such interfaces could include:

- A SQL-like interface as in PINQ and εktelo, where queries are specified as plans that are compiled into DP measurements.
- A GUI for selecting from a predefined set of statistics and transformations, as in PSI.
- A Python-like notebook interface for issuing queries to the data, like in the system we are building with Microsoft.

Another important aspect of the user interface is how accuracy and statistical confidence are exposed to users (so that analysts don't draw incorrect conclusions due to the noise for privacy), and it is important that the algorithms in the library provide support for this. We defer discussion of this to Section 5.

4.6 Contributing to the Library

We elaborate here on the different types of code contributions we envision to the library.

1. A new measurement or transformation family constructed by combining existing measurement and transformation families in the library using operators that already exist in the library (like composition, chaining, and post-processing).

   This is the easiest kind of contribution, as the certificates of privacy or stability are automatically generated by the built-in components, and we also expect it to be the most frequent contribution once the library has a large enough base of building-block components. Although no proof of privacy or stability is needed for such contributions, they will still need to be reviewed for having demonstrated utility (see the Statistics section of the OpenDP Whitepaper), code quality, and documentation.

2. A new "atomic" measurement or transformation family, where the privacy or stability properties are proven "from scratch."

   For the sake of modularity and ease of verification, such contributions should be broken down into as small and simple sub-measurements or sub-transformations as possible, each of which has a proof from scratch. Together with each such a contribution, one must also provide a mathematical proof that it actually achieves its claimed privacy or stability properties. One day, we hope that some of these proofs can be formally verified using the rapidly advancing set of tools for formal verification of differential privacy. However, in many cases, we expect that what will be provided is a written proof together with a pseudocode description of the algorithm and its claimed privacy properties. The DP researchers on the OpenDP editorial board will be responsible for verifying that the proof is correct for the pseudocode. The OpenDP Committers, with expertise on the programming framework and its implementation, will be responsible for verifying that the actual code is a faithful implementation of the pseudocode.

   Sometimes the privacy or stability proof of a new contribution will be based on some property of an existing component in the library that is not captured by the privacy or stability properties of that component. For example, "stability methods" for approximate differential privacy (like "propose–test–release") often rely on the *accuracy* of other DP mechanism (e.g. that with probability at least $1-\delta$, the Laplace mechanism produces an estimate that is correct to within $c \log(1/\delta)/\varepsilon$). Whenever this occurs, there should also be a proof provided that the said component has the given property, this proof should be attached as a certified assertion to the implementation of that component, and the privacy or stability relation for the new contribution should check the certificate. This way, if the component later changes and the assertion has to be removed, the soundness of the contribution relying on it is preserved.

3. A new combining primitive for measurements or transformations.

   This can be a new form or analysis of composition or chaining, or something more sophisticated like privacy amplification by subsampling (viewed as an operator on measurements). In the proposed framework, these are also measurement or transformation families and the process for contributing them and having them vetted and accepted is the same as above.

4. Adding a new private data type, distance measure between private data types, or privacy notion (distance measure between distributions).

   There should be a standardized format for adding such elements to the library, and what information needs to be provided and checked (again by a combination of the editorial board and committers). For example, for a new distance measure, one may provide a statement of the data types to which it applies, specify whether it satisfies properties like monotonicity and symmetry, give conversions between it and other distance measures, etc. For a new privacy notion, it needs to be verified that it satisfies the post-processing property.

5. Adding a new type of privacy or stability calculus that is not supported by the existing framework.

   Examples include introducing frameworks for privacy odometers, local sensitivity and variants (like smooth sensitivity), or per-analyst privacy budgets. These will require a more holistic architectural review before being incorporated.

4.7 The Scope of the Framework

The programming framework discussed in Section 4.2 is meant to be very general and flexible, and we believe it can support a great deal of the privacy calculus in the differential privacy literature. However, it does not support all important ways of reasoning about differential privacy. Thus, below we list some concepts that we believe can be easily supported and others that would require an expansion of the framework.

Within the Current Framework

Many different private dataset types and granularities of privacy. This includes unbounded DP (multisets, adjacency by adding or removing a record), bounded DP (datasets consisting of a known number of records, adjacency by changing a record), networks with edge-level or node-level privacy, a data stream with person-level or event-level privacy, multirelational databases with person-level privacy, DP under continual observation.

Many different measures of privacy loss. This includes pure DP, approximate DP, zero-concentrated DP, Rényi, and Gaussian DP. (We require the privacy-loss measure to be closed under postprocessing, so the original formulation of concentrated DP would not be supported.) Group privacy is also captured by the proposed privacy relations, as it allows for interrogating about the privacy loss at inputs of different distances.

Combining primitives to build complex mechanisms from simpler ones. This includes many of the sophisticated composition theorems for differential privacy (such as the advanced and optimal composition theorems for approximate differential privacy, the composition of zCDP, and more), "parallel composition" via partitioning, privacy amplification by subsampling, the sample-and-aggregate framework.

Global-sensitivity-based mechanism families. The exponential mechanism, the geometric mechanism, the (discrete) Gaussian mechanism, the Matrix Mechanism, and the Sparse Vector Technique.

Common database transformations. Subsetting according to a user-defined predicate, partitioning according to a user-defined binning, clamping into a given range/ball, vectorizing into a histogram, join (or variant) on a multi-relational dataset, user-defined row-by-row transformations, and summing over a dataset.

Restricted sensitivity and Lipschitz projections. Both of these refer to (global) stability properties of transformations with respect to rich data types, which the framework supports.

Outside the Current Framework

Below are some general concepts in differential privacy that the proposed framework does not currently support reasoning about. When these concepts are used only as an intermediate step to proving a standard privacy property, then the entire mechanism can be added to the library as a measurement. But the framework would not support reasoning directly these concepts without further augmentation.

Local sensitivity and mechanisms based on it. We envision that the local sensitivity of a transformation could be captured by adding a "local stability" relation to the attributes of a transformation, which takes a dataset as an additional argument. Such a relation would also need to be verified similarly to the already-proposed stability relations discussed in Section 4.2. By appropriate augmentations like this to the framework, we hope it will be possible to support reasoning about differentially private mechanisms based on variants of local sensitivity, such as smooth sensitivity, stability-based mechanisms, and propose-test release.

Privacy odometers. These are variants of interactive measurements that track accumulated privacy loss rather than specify a total privacy loss at the beginning [30]. As discussed in the companion paper, it should be relatively straightforward to modify the programming framework to incorporate at least pure-DP odometers.

Complex adversary models. These include pan-privacy with a limited number of intrusions, computational differential privacy, adversaries with limited collusion (e.g. when we give different analysts budgets of their own)

Randomized transformations. It is possible and sometimes useful to reason about stability properties of randomized transformations, often via couplings. For example, we can call a randomized transformation T *stable* if for every two "close" datasets x, x', there is a coupling (Y, Y') of the random variables $Y = T(x)$ and $Y' = T(x')$ such that it always holds that Y and Y' are close. The framework should be easily extendable to this way of capturing the stability properties of randomized transformations, but more complex coupling properties may be more challenging to capture.

Interactive transformations. In some DP systems, like PINQ, transformations can also be performed in an interactive and adaptive manner, creating a multitude of derived datasets (behind the "privacy barrier") on which queries can subsequently be issued. The framework proposed so far only allows for measurements to be interactive, which may mean that the same derived dataset is recomputed multiple times if it is an intermediate dataset in multiple measurements formed by chaining. This can be potentially remedied by modifying the execution engine to retain previously derived datasets and recognize when they are being recreated. However, it may eventually be useful to extend the programming framework to directly work with some form of interactive transformations, just as we allow interactive measurements. (But a general theory of interactive transformations and how they may be combined with measurements may need to be developed first.)

5 Statistics

Differential privacy is a definition of worst-case privacy loss from the release of a function on the data. Utility is an expectation of information gain from the release of the function.

The tension between these two—that information gained by the analyst is privacy lost by the subject—is foundational to the differential privacy literature. However, our present ability to reason about privacy loss outstrips our ability to speak broadly to the utility of algorithms. Every differentially private algorithm, by definition, analytically meets the privacy guarantee, but understanding of the utility to an analyst of that algorithm is often not available in analytic form (or is provided in a coarse asymptotic form that is too loose to be useful in practice). Frustratingly, utility can be dataset dependent, varying not only with privacy-loss parameters and the size of the data, but with the distribution of the data, strengths of relationships and even prior knowledge of reasonable parameter bounds. Moreover, the nature of what utility means is context and analyst dependent. The purpose of the analysis, the inferential goal (not to mention the theory of inference being used) can all radically alter how to measure the utility of a differentially private release, even from the same algorithm. Often utility results, when they exist, are presented as Monte Carlo simulation studies on a very limited range of datasets and small subsets of the possible parameters. The few results where the literature provides optimality proofs are almost always for L1 or L2 regret or error distance. This often does not match either the statistical desiderata of the analyst, nor the statistical criteria originally used to justify the estimator. In-sample, empirical risk minimization concepts of utility do not match up to the population-based inferences generally demanded of applied practitioners; neither do most DP releases provide straightforward uncertainty estimates on inferential quantities of interest.

As analysts and statisticians have begun to make practical use of differentially private releases for real world problems, the mismatch of DP releases with the inferential needs of analysts has led to a number of statements, demonstrations, and increasingly, manifestos to exactly this point [9, 23, 24, 18, 5, 6, 3, 34, 16, 15, 10].

We consider it crucial for OpenDP to try to close this gap, by bringing inferential utility to the fore as a key attribute of methods in its library of differentially private algorithms. We hope this will continue to propel the work that has been done to bridge computer science theory and applied statistics.

To this end, in the following sections we first list some of the various forms of utility that data scientists and statisticians need in practice. We then generally describe some ideas for how to make utility more transparent to users. This finally leads into a discussion of holes and needs from the applied community for differentially private tools.

5.1 Terminology

We begin with some statistics-specific terminology that typically does not make it into the privacy literature, but makes this discussion easier, and hopefully eases future conversations with applied researchers.

Data scientists, statisticians, data-driven policy makers—hereafter, for simplicity, we will collectively call these *analysts*—are all trying to reason from the data to learn something empirical. The nature of what they are trying to learn, and the specific choices that learning may lead to, all influence how to measure utility. As a general term, the thing the analyst is trying to learn can be

called the *quantity of interest*. For homophonic reasons that will shortly be obvious, this is often called the *estimand*. This might be "the fraction of children living in poverty in a region," or "the relationship between classroom size and educational attainment," or the "causal effect of a specific treatment on a patient's health outcome."

The way of measuring this quantity, that is, the function proposed to compute on the data to elicit an answer, is called the *estimator*. The estimator might perhaps be a mean, or a correlation, or a regression coefficient. There are often different plausible estimators for the same estimand, just as their there are often different DP algorithms to answer the same query.

The set of assumptions that justify why a particular estimator allows us to learn a good answer to the estimand, (including, implicitly or explicitly, the *theory of inference* that defines what learning even means), we can collectively call the *model*. The answer the estimator provides when evaluated on the data, is the *estimate*.[13]

5.2 Inference and Uncertainty

While estimators are computed on the available data, rarely is the quantity of interest itself concerned primarily with the available data. Generally, analysts are interested in learning about some larger set of persons in the world or a general principle of a repeated phenomena, of which the data at hand is merely a sample of instances. Statisticians refer to the data in hand as the *sample* and the set of possible observations one wants to learn about as the *population*.

Inference, the primary task of statistical analysis, is a set of rules that allow reasoning about the unknown population from the known sample data. Any estimate that is computed on sample data has sampling error compared to the population value. *Uncertainty* is the description of the likely magnitude of this error, and estimates that are not accompanied by expressions of their uncertainty are rarely valued by statistical analysts, as one can not measure how much one has learned about the population quantity of interest.

Estimates that are differentially private compound sampling error, with error from the noise-infusion mechanism they employ. When they exist at all, many demonstrations of DP error or utility use the sensitive sample data as the objective truth, while statisticians see even this private value as having sampling error that differs from the value in the population. Correct expressions of uncertainty describe how much can be learned about the population (not the sample) from the DP release.

5.3 Desiderata

The quantity of interest and the model assumptions collectively influence the desirable properties of the estimator and the estimate it provides. Thus the possible ways of measuring utility, that is, *how good an estimator is at providing estimates*, are broad. A brief overview of several common ones, and the circumstances in which they naturally arise is as follows.

- Error - how far the estimate is from the underlying "true" value, often expressing in Root Mean Squared Error (RMSE) which is equivalent to L2 distance.

[13] Thus to put these terms together, the inferential target of the estimator is the estimand (the thing demanded of the estimator), and the estimate is the attempt to hit that target.

- Precision/Recall - similarly, classification models have several analogs for error, including precision, recall, F1 metrics which weight between the two, and ROC curves and their area.
- Bias - the expected value of the error. Not to be confused with the expected magnitude of the error, bias measures whether estimates systematically depart from the truth in predictable directions. Researchers interested in causal reasoning typically value unbiased estimates over low error estimates.
- Variance - RMSE can be decomposed into a statement about bias and variance of the estimates. Low variance estimators are preferred in some settings.
- Coverage - When measures of uncertainty are available, and expressed as intervals (eg. confidence intervals, credible intervals) with some explicit probabilisitic definition, then the utility of these intervals is the correctness of this definition.
- Consistency - consistent estimators converge to the population value as the sample grows. Not all useful estimators are consistent, but typically inconsistent estimators need some additional justification to overcome the unease that inconsistency provokes.
- Invariance - estimates of quantities of interest should not be dependent on scaling choices, or other arbitrary identification choices.
- Power - how small an effect can be observed with a finite sample of data.

5.4 Tracking, Recording and Use of the DGP

An important advantage of differential privacy for secondary statistical use is that the entire noise addition process is known. In many SDL contexts, parts of the noise mechanism have to themselves be kept obscure. The knowledge of the noise addition allows that to be incorporated into any secondary model using differentially private outputs as inputs. For example, the noise mechanism can be included into a likelihood function, or a measurement error correction, or the numerical simulation of test statistics, or any other part of a statistical model using the data generating process (DGP). Utilities for tracking the total noise addition, or specifying the exact noise distribution as well as standards for attaching these as metadata to releases would allow higher utility reuse of these releases by analysts.

5.5 Benchmarking Strategy

When analytical results are not feasible for differentially private mechanisms, simulation studies provide useful ways to judge which DP estimator will work well for a particular estimand. However, utility often varies based on a very large space of parameters, including the size and dimension of the data, the data types and distribution, strengths of relationships and even prior knowledge of reasonable parameter bounds. In addition, as noted above there are a large set of context specific utility metrics. Papers can never fully detail this entire choice space, and it is a heavy workload for authors to even try to explore the space of possible datasets. A infrastructure for benchmarking, as for example developed with [21], where new additions to the library can be tested across a large corpus of datasets and parameter specifications and utility metrics would be useful to authors and

users. Also, algorithms that require certain parameters to be first tuned on similarly structured pubic data before proceeding with a private analysis (such as for example [1]), could use this corpus and infrastructure as an explicit part of their analysis.

Bench marking against public data can provide *a priori* utility measures that can try to convey the utility of an analysis, before the analysis is conducted. This is particularly useful to help the analyst decide whether it worth running a given DP algorithm on their data, or whether the result is likely to be a useless waste of privacy-loss budget. The measures of uncertainty described in section 5.2 would predominantly be *a posteriori* utility measures, calculated after the analysis takes place, possibly itself consuming some privacy budget, and able to convery the uncertainty of the final analysis releases.

5.6 Propositions

From this discussion, the following are possible propositions for the OpenDP community.

1. To facilitate applied research, the OpenDP library should immediately prioritize incorporating a collection of algorithms that provide uncertainty estimates, analytical error promises, and data generating processes that are easy to incorporate into statistical models.

2. Analysts need different categories of algorithms, and a priority should be to round out methods in each of these. They need 1] general purpose estimators (for example, such as subsample-and-aggregate) that allow broad, flexible querying of the data, 2] DP optimizers that allow flexibility in model construction, but also will require 3] performance optimized versions of commonly used estimators (for example, linear regression, difference-of-means). To begin the conversation, a list of possible estimators, separated by these categories, is included below.

3. There is not one single or unifying utility measure. Library components should support different systems of inference and notions of utility corresponding to different inferential objectives. How should utility be exposed within the library programming framework?

4. Contributions to the OpenDP library ought to come with some measure or evidence of utility. At the least, this might be a written note describing experimental results. These utility results might be categorized into broad categories such as "descriptive validity", "statistical validity" with experimental and analytic variants.

5. The OpenDP platform should provide automated support for Monte Carlo experiments on public datasets (e.g. from Dataverse) or synthetic data. Infrastructure for automated benchmarking should be a priority of the OpenDP Commons. Easy benchmarking both facilitates applied users understanding, and provides an incentive for code contributions to get access to this utility.

5.7 Statistics Prioritization List

To promote discussion, we provide below a DP statistics wishlist that should be prioritized for inclusion into the OpenDP library to facilitate statistical workflows. These are arranged by categories within a typical workflow. Most of these are not closed problems, and some currently have no practical general purpose DP algorithm.

- Data Curation
  - Methods for metadata creation
- Data Cleaning and Preprocessing
  - Outlier detection
  - Missing data treatment and imputation
  - Matching methods
- Exploratory Data Analysis
  - Visualization
  - Histograms
  - Univariate Statistics
  - Medians, Quantiles, Range Queries
  - PCA
- Modelling
  - Generalized Optimization Methods
  - Sample-and-Aggregate approaches
  - Linear and Logistic regression
  - single dependent variable likelihood methods: GLM-like, count models
- Hypothesis Testing
  - Difference-of-means
  - Tests of Independence
- Synthetic Data Generation

6 Collaborations

We refer readers back to Figure 1 in Section 1 where we describe how the OpenDP Commons provides community governed components shared across numerous end-to-end systems with their own additions, adaptations and governance, oriented around individual needs and use cases. We anticipate that these end-to-end systems may be research platforms written by academics, or data portals for repositories or government agencies, or indeed industry products with commercial markets. For the OpenDP Commons to remain vibrant and valuable to system builders requires a healthy interchange between contributors to the Commons and system builders.

Our goal is to make sure that researchers and developers contributing to the OpenDP Commons are supported for their effort, and rewarded with credit for the contribution and impact. Similarly, we want to make sure that the components built in the Commons are guided by the needs of system builders.

Most directly, we propose an industry support model, where corporations (from giant to startup) can contribute financially, and in return both brand their products as using this authoritative, trusted, vetted codebase. Additionally, industry sponsors would gain increased representation to guide issues and priorities for component development.

Just as importantly, we anticipate that one crucial function OpenDP as an institution can provide is to be a "match-maker" between system builder needs and researcher expertise. System builders who want to support research in a topic they require, rather than needing to run their own grant making apparatus, searching for candidates, and evaluating expertise, can use the OpenDP committees to find OpenDP Community members who can provide their expertise in return for research funding. Similarly, community members who have novel research contributions should be able to use the OpenDP committees to find either industry partners to bring their work quickly to practice, or applied researchers who can translate their innovations into high impact use cases. For example, researchers in the differential privacy community have a strong desire to maximize the impact of their work, and OpenDP can provide a channel for such impact. We envision that OpenDP will play a "match-maker" by aligning specific functional needs of OpenDP users with capabilities of external DP experts to develop and contribute the appropriate DP solutions to OpenDP. The DP Application Leader(s) role is created specifically for this purpose, and we aim to create systems for match-making and grant distribution immediately, to foster community integration.

In the longer term, we have several ways we want to put the OpenDP community and platform at the heart of both academic research and industry adoption. Another way of maximizing incentives for OpenDP development is to provide code contributors with measures of how widely their code is used, and encourage users of the library to cite the researchers who designed the algorithms they used.

Along these lines, we propose to make the OpenDP libraries self-citing, so that as an analysis is constructed, all the algorithms that have been used are put into a bibliography that credits the original paper authors, to facilitate the proper citation of works contributed to the library. The cloud deployment of the library in the OpenDP system could also track the usage of algorithms and report metrics to the original authors to show the impact of their work. Another idea is to offer automated benchmarking against a corpus of test datasets (building on DPBench [21]), so that contributors of algorithms can easily demonstrate the utility of their contributions in papers they produce from their work. A "bug bounty" program can be used to incentivize the community to carefully inspect any new contributions to OpenDP for vulnerabilities.

To foster contributions from and use by industry, we envision offering six-month residencies, wherein companies could send developers to join the OpenDP team, contribute to OpenDP codebase, and learn and train in differential privacy before returning to their companies, where they may also continue to contribute to the OpenDP codebase as their company makes use of OpenDP's tools. (The Harvard Privacy Tools project, has had extensive experience in teaching differential privacy through the tutorials we have offered every year in the very successful Privacy Tools summer internship program, and the "Applied Privacy for Data Science" graduate course developed and taught by Honaker and Vadhan. We would anticipate expanding these offerings and drawing in more community members for workshop, summer training, and other outreach and pedagogical opportunities.)

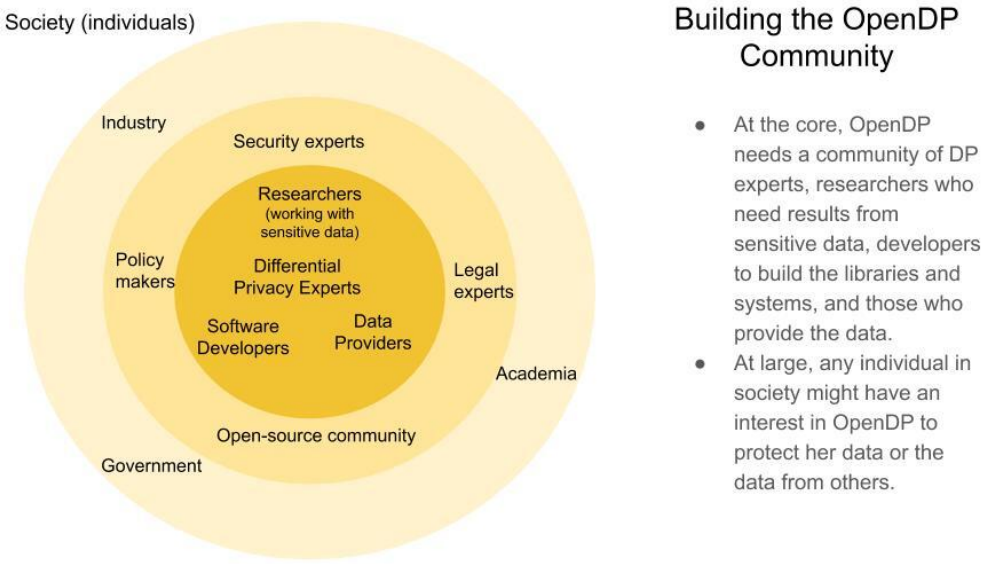

Figure 4: *Definition of OpenDP Community*

7 System Integration

For differential privacy software to be useful in applications, it must integrate with end-to-end systems, which may vary in their back-end storage and compute capabilities and in the needs and expertise of end users. In this whitepaper, we give several examples of the kinds of systems we aim to support with OpenDP, including the forthcoming system developed in collaboration with Microsoft, and the challenges that each of them raise.

We have core design principles which we would like to use to assure that we build the most trusted and capable systems possible. Open source, privacy and security, scalability and extensibility are paramount to our designs.

Our work will be released via an open source license. There are lots of choices inherent in any actual deployment of differential privacy to a particular use case—choices in architecture, algorithms, trade-offs between ease of use versus flexibility, even choices of what types of data to support. Our goal is to harness the knowledge and abilities of the community of experts to make decisions reflecting the state of the art, while maintain flexibility when relative trade-offs favor different styles of applications.

We put privacy and security first. Changes to the differentially private algorithms in our libraries will be vetted by experts in differential privacy, Any additions to the API or to any critical component of the service will be vetted by security experts.

Scalability is a first class citizen in our architectural decision making. As our systems change and grow, we recognize the fact that useful code which does not perform at scale has scant utility for the overall community. We weigh scalable capabilities more heavily than those which do not scale.

We write reusable software. Our systems must be written in a modular fashion, so that they are lasting, modifiable, and of maximum utility for future systems.

We want to architect OpenDP so that the system can be adapted in the longer run to have broad adoption and use, while not needlessly slowing immediate application to the currently prioritized use cases.

7.1 Early OpenDP Systems

The forthcoming system with Microsoft is an example of an end-to-end system that OpenDP can facilitate. It works on the *curator* model, previously described in section 2.4. At the core is a library of components, operators and differentially private algorithms. These components are managed in a directed acyclic graph data structure. If a user constructs a query made out of these components, our validator certifies that the proposed workflow meets the differential privacy definition, and that the total information leakage (privacy loss) is within the specified limit. If these conditions are met, then three processes occur. The data is passed to the privacy module, the analysis is computed using our runtime, and the privacy-preserving answer is returned to the user.

The graph of library components is used to perform an analysis on the data. This graph, defined through a Protocol Buffers API, is statically validated. This validation proves out the privacy preserving properties of the sequence, without ever compromising the sensitivity of the underlying data. If privacy is guaranteed, then the analysis is executed in a Rust-implemented runtime.

This system will be launched in the spring of 2020.

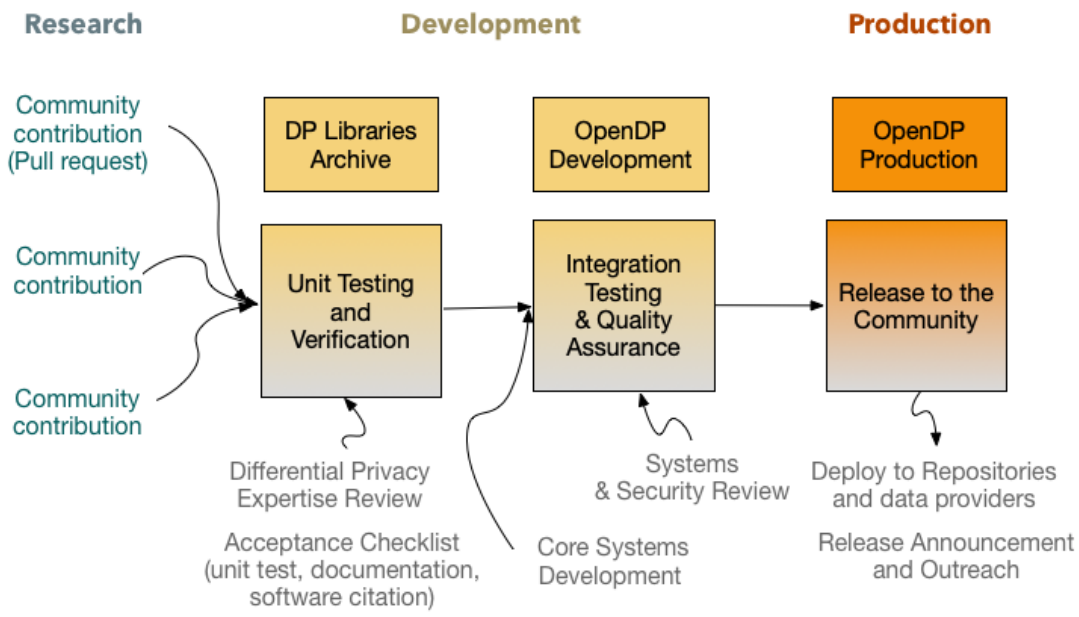

Figure 5: *OpenDP product lifecycle.*

The key elements of the system are:

1. **Library of DP Algorithms:** The mechanisms library provides a fast, memory-safe native runtime for validating and running differentially private analyses. The runtime and validator are built in Rust, while an additional language binding for Python is available. A binding in R is forthcoming. Mechanisms and Algorithms are pluggable. We currently provide Gaussian, Laplace, and Geometric algorithms, and common statistics such as Count, Sum, Mean, Covariance.

2. **Analysis Graph API:** Differentially private computations are specified as a computational graph that defines a proposed analysis. This plan can be validated and executed to produce differentially private releases of data. Releases include metadata about accuracy of outputs and the complete privacy cost of the analysis.

3. **SDK:** The data access SDK provides an API to produce differentially private reports from SQL queries. The API is designed to functions in a similar way as to an online database connectivity connection (ODBC). We support a subset of SQL-92, focused on common aggregates used in analytics. Queries are mapped to an analysis graph, and can be executed against a variety of back-ends, including CSV files, simple database tables, and data sets related to Dataverse.

4. In the future, the SDK may include a sample **MLFlow execution environment**. This environment is intended to demonstrate how to compose a query budget from multiple types of differentially private processes into a single service.

5. **Static Validator:** The static validator can take a proposed analysis, defined as a computational graph in the API, and determine whether the proposed releases all meet the differential

privacy definition, and a worst-case value for cumulative privacy loss.

6. **Stochastic Validator:** The stochastic evaluator automatically tests any black-box differential privacy algorithm to look for potential violations of privacy or accuracy properties. Samples for testing the native runtime and data access SDK are provided.

7.2 Data Repositories

Data repositories are a key part of the differential privacy ecosystem which we envision for OpenDP. Many offer critical insights, but fall short in their ability to share sensitive data while retaining privacy. The sensitive data sets are commonly locked away in archives, or shared using ineffective "de-identification" methods that fail to protect privacy. Lengthy human-intensive release processes introduce inefficiencies to timely data sharing. We feel that data repositories are a critical area in which differentially private data release could help.

Dataverse. Dataverse is a an open-source software platform for building data repositories. At the time of this writing, there are 56 Dataverse repositories in production world-wide, each one hosting data for one or many organizations or academic institutions. Dataverse has the ability to access storage locally and remotely. The current plan for Dataverse integration with OpenDP includes leveraging two other project, the DataTags system () and ImPACT (https://cyberimpact.us/). The DataTags system helps classify the data deposited or discoverable through Dataverse into 6 levels (blue, green, yellow, orange, red, crimson) based on how sensitive the dataset or data files are, and the access and security requirements. Datasets with orange, red, and crimson datatags need to be stored in a remote trusted storage, and can only be accessed through a notary service. Dataverse plans to use components from the project ImPACT to support the storage and access of sensitive data. OpenDP will be integrated with Dataverse sensitive datasets as an external tool to run statistical queries on those datasets, without the end user ever accessing the raw data. The OpenDP system that integrates with Dataverse will be able to access the data in ImPACT's Trusted Remote Storage Agents (TRSA) through the Notary Service. This proposed integration is open for review and needs to be detailed based on Dataverse use cases.

7.3 Key Components to Consider for OpenDP Systems

7.3.1 Identity, Authentication and Authorization

Our initial intention is to have OpenDP system identity and access enforcement be role-based. Suggested roles within the analyst organization include privacy officer, identity administrator, statistician/data depositor and data administrator. The statistician/data depositor is well-versed in statistics to the point that they can responsibly choose a privacy loss parameter value, choose types and ranges for variables in the release, select an initial set of differentially private statistics to calculate and release, and determine how the remaining privacy budget will be partitioned. Roles within the hosting organization are similar, though the privacy officer role is not required. This omission is due to the ownership of the data, which belongs to the analyst organization.

We are actively seeking additional guidance on roles that will best accommodate privacy stakeholders.

ORCID. We are considering ORCID for our initial identity life cycle management system. Orcid provides persistent digital identifiers to researchers. Access to ORCID will be via the latest stable version of OAuth. For organizations which find an IDaaS manager undesirable, locally hosted identities hosted via an LDAP protocol-based system will be accepted.

We seek to understand the identity scenarios anticipated by our stakeholders. In particular we would like to learn about non-enterprise identity needs.

Role Enforcement. Role enforcement is envisioned to be managed first via a simple configuration file within the system, and later via an LDAP protocol-based system. More complex IAM systems such as AWS Identity and Access Management appear to be out of scope for early OpenDP systems. If stakeholders find that basic role configuration is inadequate for the capabilities they envision, we are willing to reconsider this limitation. We actively seek user stories which highlight requirements related to role enforcement.

7.3.2 Data Access

Local File. We have begun work on one OpenDP system joint with Microsoft soon to be released. The system is a tool allowing researchers and analysts to create and query formerly private data that has undergone differentially private releases. The system will focus on stochastic testing of the modules, and a user experience via Jupyter Notebooks and a SQL-like language interface. The Rust-based engine which produces the library is flexible enough to create libraries in a multitude of target languages. The initial release offers the library in Python.

We plan that the first OpenDP system will accept data in CSV format, uploaded directly from the client system. Files will be stored locally, owned and managed by the organization hosting the OpenDP system. This limited choice of data upload options is based upon limitations currently in place on the core library. In the project joint with Microsoft, being our initial OpenDP system, the user experience which handles more advanced capabilities like uploading relational tables is not currently connected to the core library.

OpenDP systems will initially support datasets that are simple flat tables where the unit of observation is an individual data subject. However, many more complex forms of data require their own tailored algorithms (and even privacy definitions). We would plan to extend the system to the following types of data, prioritized based on the interests of the community of researchers and the needs of the use cases, practitioners, and industry partners who join our community.

Relational Database. In order to discourage the exploitation of security vulnerabilities, we propose access to relational databases such as Oracle, SQL Server, MySQL and DB2 be made indirect, via REST API provided by the database-owning organization. We propose that no direct ODBC connection be initially available among OpenDP systems and relational data stores. We are currently seeking feedback on the tradeoffs between security and greater access to data storage systems. We are proposing that OpenDP systems requiring direct access between the system and a relational database be placed as architecturally close to the database as is possible. This proximity is intended to minimize attack surface. We are further considering the limitation that OpenDP projects unable to gain database access via a REST API or direct integration will use a secured ODBC driver to access the database.

We seek feedback on these measures, which are intended to ensure that as a privacy tool, an OpenDP system avoids introducing vulnerabilities.

NoSQL Database. NoSQL databases, including key-value stores such as Redis, document stores like Elastic, and graph databases including TopQuadrant and AWS Neptune often guarantee neither atomicity nor consistency. For this reason, OpenDP systems may be mathematically and architecturally challenged to directly integrate with NoSQL databases. For NoSQL databases which use a normalization strategy to avoid duplication of data, we envision that architectural integration will be similar to that of relational databases, as described elsewhere in this document. For de-normalized NoSQL databases, inconsistent and duplicated data pose hazards which make such databases poor candidates for mathematically provable differential privacy. However, in our Microsoft collaboration, our initial user experience MVP includes access to such NoSQL databases, for purposes of gauging the utility it provides to prospective users. As technology is constantly evolving, our goal is to revisit limitations placed on OpenDP systems in order to avoid missed opportunities for capability expansion.

Resilient Distributed Datasets. Resilient distributed datasets, including Apache Spark, will initially only be accessible via REST API provided by the product itself or by the distributed dataset-owning organization. As requirements evolve, this architecture may be reviewed. Concerns about atomicity and consistency weigh equally upon resilient distributed datasets as they do upon NoSQL databases (see above).

7.3.3 Application Programming Interfaces

OpenDP systems will be accessible to other parts of the application architecture via APIs. As with other aspects of system integration, the issues of privacy, security, scalability and extensibility will have a strong influence over API design.

Microservice Architecture. Beyond these critical areas, the design of OpenDP system APIs would benefit from being maintainable and testable, loosely coupled with other services, independently deployable, and able to be worked on by an API development team without disturbing core application development. These services based on the microservice architecture pattern[14] can be further enhanced by packaging as part of a gateway.

API Gateway. During the lifecycle of an application, the needs of clients change. The granularity, data format and network performance needs can transform enough that existing services are rendered inadequate. Technical details like the protocol of the microservice, the number of microservice instances, and the composition into greater services can become sources of technical debt. One solution to many of these issues is to apply the API gateway pattern[15]. An API gateway provides a single point of entry for all clients. The API gateway routes requests to the appropriate

[14] https://martinfowler.com/articles/microservices.html
[15] https://microservices.io/patterns/apigateway.html

microservice, or sending requests out to multiple back-end services. The variation in external requests is encapsulated within the API gateway, preventing complexity and technical debt related to client request changes from affecting the implementation of core functionality.

Appendices

A What is Differential Privacy?

[This section, except for the fifth paragraph, is a verbatim extract from the differential privacy primer [35], coauthored by Executive Committee members Honaker, Vadhan, and collaborators.]

Differential privacy is a strong, mathematical definition of privacy in the context of statistical and machine learning analysis. It is used to enable the collection, analysis, and sharing of a broad range of statistical estimates based on personal data, such as averages, contingency tables, and synthetic data, while protecting the privacy of the individuals in the data.

Differential privacy is not a single tool, but rather a criterion, which many tools for analyzing sensitive personal information have been devised to satisfy. It provides a mathematically provable guarantee of privacy protection against a wide range of privacy attacks, defined as attempts to learn private information specific to individuals from a data release. Privacy attacks include re-identification, record linkage, and differencing attacks, but may also include other attacks currently unknown or unforeseen. These concerns are separate from security attacks, which are characterized by attempts to exploit vulnerabilities in order to gain unauthorized access to a system.

Computer scientists have developed a robust theory for differential privacy over the last fifteen years, and major commercial and government implementations are starting to emerge. Differential privacy mathematically guarantees that anyone viewing the result of a differentially private analysis will essentially make the same inference about any individual's private information, whether or not that individual's private information is included in the input to the analysis.

What can be learned about an individual as a result of her private information being included in a differentially private analysis is limited and quantified by a privacy loss parameter, usually denoted epsilon (ε). Privacy loss can grow as an individual's information is used in multiple analyses, but the increase is bounded as a known function of ε and the number of analyses performed.

Differentially private algorithms are constructed by carefully introducing "random noise" into statistical analyses so as to obscure the effect of each individual data subject. Thus, differential privacy reduces the accuracy of statistical analyses, but does so in a quantifiable manner that introduces an explicit privacy-utility tradeoff. As the number n of observations in a dataset grows sufficiently large, the loss in accuracy due to differential privacy generally becomes much smaller than that due to statistical sampling error. However, it can be challenging to maintain high accuracy for studies on modest-sized datasets (or modest-sized subsets of large datasets).

The differential privacy guarantee can be understood in reference to other privacy concepts:

- Differential privacy protects an individual's information essentially as if her information were not used in the analysis at all, in the sense that the outcome of a differentially private algorithm is approximately the same whether the individual's information was used or not.
- Differential privacy ensures that using an individual's data will not reveal essentially any personally identifiable information that is specific to her, or even whether the individual's information was used at all. Here, specific refers to information that cannot be inferred unless the individual's information is used in the analysis.

As these statements suggest, differential privacy is a new way of protecting privacy that is more quantifiable and comprehensive than the concepts of privacy underlying many existing laws,

policies, and practices around privacy and data protection. The differential privacy guarantee can be interpreted in reference to these other concepts, and can even accommodate variations in how they are defined across different laws. In many cases, data holders may use differential privacy to demonstrate that they have complied with legal and policy requirements for privacy protection.

Differential privacy is currently in initial stages of implementation and use in various academic, industry, and government settings, and the number of practical tools providing this guarantee is continually growing. Multiple implementations of differential privacy have been deployed by corporations such as Google, Apple, and Uber, as well as federal agencies like the US Census Bureau. Additional differentially private tools are currently under development across industry and academia.

Some differentially private tools utilize an interactive mechanism, enabling users to submit queries about a dataset and receive corresponding differentially private results, such as custom-generated linear regressions. Other tools are non-interactive, enabling static data or data summaries, such as synthetic data or contingency tables, to be released and used. In addition, some tools rely on a curator model, in which a database administrator has access to and uses private data to generate differentially private data summaries. Others rely on a local model, which does not require individuals to share their private data with a trusted third party, but rather requires individuals to answer questions about their own data in a differentially private manner. In a local model, each of these differentially private answers is not useful on its own, but many of them can be aggregated to perform useful statistical analysis.

Differential privacy is supported by a rich and rapidly advancing theory that enables one to reason with mathematical rigor about privacy risk. Adopting this formal approach to privacy yields a number of practical benefits for users:

- Systems that adhere to strong formal definitions like differential privacy provide protection that is robust to a wide range of potential privacy attacks, including attacks that are unknown at the time of deployment. An analyst using differentially private tools need not anticipate particular types of privacy attacks, as the guarantees of differential privacy hold regardless of the attack method that may be used.

- Differential privacy provides provable privacy guarantees with respect to the cumulative risk from successive data releases and is the only existing approach to privacy that provides such a guarantee.

- Differentially private tools also have the benefit of transparency, as it is not necessary to maintain secrecy around a differentially private computation or its parameters. This feature distinguishes differentially private tools from traditional de-identification techniques, which often conceal the extent to which the data have been transformed, thereby leaving data users with uncertainty regarding the accuracy of analyses on the data.

- Differentially private tools can be used to provide broad, public access to data or data summaries while preserving privacy. They can even enable wide access to data that cannot otherwise be shared due to privacy concerns. An important example is the use of differentially private synthetic data generation to produce public-use microdata.

Differentially private tools can, therefore, help enable researchers, policymakers, and businesses to

analyze and share sensitive data, while providing strong guarantees of privacy to the individuals in the data.

B The MIT License

Source: https://opensource.org/licenses/MIT

Copyright ⟨YEAR⟩⟨COPYRIGHT HOLDER⟩

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

C Contributor License Agreement

Form Contributor License Agreement w/Company Consent (DRAFT)
CONTRIBUTOR LICENSE AGREEMENT

This Contributor License Agreement (the "CLA") is entered into by and between President and Fellows of Harvard College, ("Harvard"), and [NAME] ("Contributor"), in connection with Contributor's participation in the OpenDP initiative at Harvard University.

1. Contributor accepts and agrees to the terms and conditions of this CLA, which govern any present or future original work of authorship (including but not limited to modifications or additions to an existing work) intentionally submitted by Contributor to Harvard in connection with the OpenDP initiative, for inclusion in, or documentation of, any product owned or managed by Harvard (the "Work").

2. Except for the license granted herein to Harvard and recipients of software distributed by Harvard, Contributor reserves all right, title, and interest in and to Contributions, including but not limited to any copyrights therein.

3. Subject to the terms and conditions of this CLA, Contributor grants to Harvard and to recipients of software distributed by Harvard a perpetual, irrevocable, non-exclusive, worldwide, royalty-free license to reproduce, prepare derivative works of, publicly display, publicly perform, sublicense, and distribute Contributions and such derivative works.

4. Subject to the terms and conditions of this CLA, Contributor grants to Harvard and to recipients of software distributed by Harvard a perpetual, irrevocable, non-exclusive, worldwide, royalty-free patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work. Such license applies only to those patent claims licensable by Contributor that are necessarily infringed by Contributor's Contribution(s) alone or by combination of Contributor's Contribution(s) with the Work to which such Contribution(s) was submitted. If any entity institutes patent litigation against Contributor or any other entity (including a cross-claim or counterclaim in a lawsuit) alleging that your Contribution, or the Work to which Contributor has contributed, constitutes direct or contributory patent infringement, then any patent licenses granted to that entity under this Agreement for that Contribution or Work shall terminate as of the date such litigation is commenced.

5. Contributor represents that Contributor is legally entitled, has the legal authority and has all rights necessary to grant the license in this CLA. If Contributor's employer(s) has rights to intellectual property that Contributor creates, including rights to Contributions, Contributor represents that:

   (a) Contributor has received permission to make Contributions on behalf of that employer, as evidenced by an authorized signature on the accompanying Company Consent Form; and

   (b) Contributor's employer has waived such rights with respect to Contributions.

6. Contributor represents that each Contribution is Contributor's original creation.

7. Contributor is not expected to provide support for Contributor's Contributions, except to the extent Contributor desires to provide such support. Contributor may provide support for free, for a fee, or not at all. Unless required by applicable law or agreed to in writing, Contributor provides Contributions on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE.

8. Contributor agrees to notify Harvard of any facts or circumstances of which Contributor becomes aware that would make any representations set forth herein inaccurate in any respect.

CONTRIBUTOR		HARVARD	
Print Name:		Print Name:	
Date:		Date:	

COMPANY CONSENT FORM

[NAME], an employee of [EMPLOYER] has requested to participate in the OpenDP initiative at Harvard University.

In order to participate, the Employee will be required to execute a Contributor License Agreement, a copy of which is attached. The terms and conditions of that agreement are incorporated by reference and made a part of this form. By signing below, the Company agrees that the Employee's Contributions, whether such Contributions are owned by the Company or Employee, may be intentionally submitted to Harvard in connection with the OpenDP initiative, in accordance with the terms of the Contributor License Agreement.

By signing below, you are representing that you have the authority to provide this consent on behalf of the Company.

Please sign: _____ Date: _____
Name: _____
Corporation: _____

References

[1] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, 2016.

[2] M. Altman, A. Wood, D. R. O'Brien, S. Vadhan, and U. Gasser. Towards a modern approach to privacy-aware government data releases. *Berkeley Technology Law Journal*, 30(3):1967–2072, 2015.

[3] J. Awan and A. Slavković. Differentially private uniformly most powerful tests for binomial data. In *Advances in Neural Information Processing Systems*, pages 4208–4218, 2018.

[4] V. Balcer and S. Vadhan. Differential Privacy on Finite Computers. *Journal of Privacy and Confidentiality*, 9(2), September 2019. Special Issue on TPDP 2017. Preliminary versions in ITCS 2018 and posted as arXiv:1709.05396 [cs.DS].

[5] T. Brawner and J. Honaker. Bootstrap inference and differential privacy: Standard errors for free. *Unpublished Manuscript*, 2018.

[6] Y. Chen, A. Machanavajjhala, J. P. Reiter, and A. F. Barrientos. Differentially private regression diagnostics. In *ICDM*, pages 81–90, 2016.

[7] M. Crosas. The dataverse network®: an open-source application for sharing, discovering and preserving data. *D-lib Magazine*, 17(1):2, 2011.

[8] M. Crosas. A data sharing story. *Journal of eScience Librarianship*, 1(3):7, 2013.

[9] V. D'Orazio, J. Honaker, and G. King. Differential privacy for social science inference. *Sloan Foundation Economics Research Paper*, (2676160), 2015.

[10] W. Du, C. Foot, M. Moniot, A. Bray, and A. Groce. Differentially private confidence intervals. *arXiv preprint arXiv:2001.02285*, 2020.

[11] C. Dwork, V. Feldman, M. Hardt, T. Pitassi, O. Reingold, and A. Roth. Generalization in adaptive data analysis and holdout reuse. In *Advances in Neural Information Processing Systems*, pages 2350–2358, 2015.

[12] C. Dwork, V. Feldman, M. Hardt, T. Pitassi, O. Reingold, and A. Roth. The reusable holdout: Preserving validity in adaptive data analysis. *Science*, 349(6248):636–638, 2015.

[13] C. Dwork, V. Feldman, M. Hardt, T. Pitassi, O. Reingold, and A. Roth. Guilt-free data reuse. *Communications of the ACM*, 60(4):86–93, 2017.

[14] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[15] G. Evans and G. King. Statistically valid inferences from differentially private data releases. Manuscript, 2020.

[16] G. Evans, G. King, M. Schwenzfeier, and A. Thakurta. Statistically valid inferences from privacy protected data. Manuscript, 2019.

[17] M. Gaboardi, M. Hay, and S. Vadhan. A programming framework for opendp. Manuscript, May 2020.

[18] M. Gaboardi, J. Honaker, G. King, J. Murtagh, K. Nissim, J. Ullman, and S. Vadhan. Psi (ψ): a private data sharing interface. *arXiv preprint arXiv:1609.04340*, 2016.

[19] S. L. Garfinkel. Draft (2nd) nist sp 800-188, de-identification of government datasets, 2016.

[20] A. Haeberlen, B. C. Pierce, and A. Narayan. Differential privacy under fire. In *USENIX Security Symposium*, volume 33, 2011.

[21] M. Hay, A. Machanavajjhala, G. Miklau, Y. Chen, and D. Zhang. Principled evaluation of differentially private algorithms using dpbench. In *Proceedings of the 2016 International Conference on Management of Data*, pages 139–154, 2016.

[22] N. M. Johnson, J. P. Near, J. M. Hellerstein, and D. Song. Chorus: Differential privacy via query rewriting. *CoRR*, abs/1809.07750, 2018.

[23] V. Karwa, D. Kifer, and A. B. Slavković. Private posterior distributions from variational approximations. *arXiv preprint arXiv:1511.07896*, 2015.

[24] V. Karwa and S. Vadhan. Finite sample differentially private confidence intervals. *arXiv preprint arXiv:1711.03908*, 2017.

[25] G. King. An introduction to the dataverse network as an infrastructure for data sharing, 2007.

[26] G. King. Restructuring the social sciences: reflections from harvard's institute for quantitative social science. *PS: Political Science & Politics*, 47(1):165–172, 2014.

[27] F. D. McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, pages 19–30, 2009.

[28] I. Mironov. On significance of the least significant bits for differential privacy. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 650–661, 2012.

[29] K. Nissim and U. Stemmer. On the generalization properties of differential privacy. *CoRR, abs/1504.05800*, 2015.

[30] R. Rogers, A. Roth, J. Ullman, and S. Vadhan. Privacy odometers and filters: Pay-as-you-go composition. In *Advances in Neural Information Processing Systems 29 (NIPS '16)*, pages 1921–1929, December 2016. Full version posted as arXiv:1605.08294 [cs.CR].

[31] L. Sweeney. Weaving technology and policy together to maintain confidentiality. *The Journal of Law, Medicine & Ethics*, 25(2-3):98–110, 1997.

[32] B. Ubaldi. Open government data: Towards empirical analysis of open government data initiativesubaldi, b.(2013). *Open Government Data: Towards Empirical Analysis of Open Government Data Initiatives. Oecd,(22). Retrieved from www. oecd. org/daf/inv/investment-policy/. Oecd. Epub ahead of print*, 10, 2013.

[33] N. Veljković, S. Bogdanović-Dinić, and L. Stoimenov. Benchmarking open government: An open data perspective. *Government Information Quarterly*, 31(2):278–290, 2014.

[34] Y. Wang, D. Kifer, J. Lee, and V. Karwa. Statistical approximating distributions under differential privacy. *Journal of Privacy and Confidentiality*, 8(1), 2018.

[35] A. Wood, M. Altman, A. Bembenek, M. Bun, M. Gaboardi, J. Honaker, K. Nissim, D. R. O'Brien, T. Steinke, and S. Vadhan. Differential privacy: A primer for a non-technical audience. *Vand. J. Ent. & Tech. L.*, 21:209, 2018.

[36] D. Zhang, R. McKenna, I. Kotsogiannis, M. Hay, A. Machanavajjhala, and G. Miklau. Ektelo: A framework for defining differentially-private computations. In *Proceedings of the 2018 International Conference on Management of Data*, pages 115–130, 2018.

[37] A. Zuiderwijk and M. Janssen. Open data policies, their implementation and impact: A framework for comparison. *Government Information Quarterly*, 31(1):17–29, 2014.