List of definitions used in the proofs

Sílvia Casacuberta, Grace Tian, and Connor Wagaman

Summer 2021
Version as of September 6, 2021 (UTC)

We use the following guideline: if a term appears in the `\texttt{...}` LaTeXfont, then this term is defined in the pseudocode definitions document. Otherwise, it appears in the proof definitions document.

We provide the terms in alphabetical order within each section. "TODOs" should be included at the end of the corresponding section. On the other hand, "TODOs" which better specify an already-defined term should be included immediately following the definition of that term. Examples should never be part of the definition, but we encourage their use right after the definition of a term.

Note that, as of September 6, 2021, the definitions in sections 5+ are not used in the most finalized proof documents. Please leave feedback, but note that the correctness of those sections does not impact the correctness of the proofs that are most finalized.

Contents

0.1 List of terms that have not yet been added

- Generalization of the path property and the corresponding proofs
- Better definition for multisets, and an explanation of the bijection between multisets and histograms
- Clarification sentence on the stability relation
- Approach with floating point. New from 3/8: explain the different rounding modes in MPFR and which is the preferred one (round to nearest). Assume idealized model for sampling and the discretized versions. For now: this is in another Overleaf document: `https://www.overleaf.com/project/611159043230572a988ee69c`.
- Explain unification of vector and scalar versions in the case of L1 distance (e.g., Laplace).

0.2 Versions of definitions documents

Other definitions documents are linked here for convenience. This definitions document is not dependent on other definitions documents, but some implementation-related details can be found in the documents listed below.

- **Pseudocode definitions document:** can be found at this link (as of September 6, 2021)

1 Mathematical operators

The notation we are using for various mathematical operations is[1] inspired by section 2 of `https://hal-ens-lyon.archives-ouvertes.fr/ensl-01529804/file/crlibm.pdf`.

[1] As of June 24, 2021.

(The notation may change in the future.) We plan to use a similar standardized notation for describing the semantics of MPFR.[2]

Definition 1.1 $(+, -, \times)$**.** The symbols $+$, $-$, and $\times$ represent the usual mathematical operations of addition, subtraction, and multiplication, respectively.

Definition 1.2 $(\oplus, \ominus, \otimes)$**.** The symbols $\oplus$, $\ominus$, and $\otimes$ denote the corresponding operations on the associated Rust type (rather than on the real numbers).

Definition 1.3 $(\rightarrow)$**.** Let $\mathcal{X}$ and $\mathcal{Y}$ be domains. For the deterministic function $f : \mathcal{X} \rightarrow \mathcal{Y}$, the arrow $\rightarrow$ represents a deterministic mapping from domain $\mathcal{X}$ to domain $\mathcal{Y}$.

Definition 1.4 $(\rightsquigarrow)$**.** Let $\mathcal{X}$ and $\mathcal{Y}$ be domains. For the randomized function $f : \mathcal{X} \rightsquigarrow \mathcal{Y}$, the squiggly arrow $\rightsquigarrow$ represents a randomized mapping from domain $\mathcal{X}$ to domain $\mathcal{Y}$.

1.1 Notes, todos, questions

2 Data Representation

Definition 2.1 (Vector)**.** A vector v is an ordered list of objects.

A vector is not necessarily Rust vector, but an idealized data container with some notion of records and ordering.

Definition 2.2 (Set)**.** A set is an unordered list of objects.

Definition 2.3 (Multiset)**.** A multiset is a modification of the notion of a set which, unlike a set, allows for repetitions for each of its elements. The number of repetitions of an element in the multiset corresponds to the *multiplicity* of that element.

Remark 2.4. We use the notation $\mathrm{MultiSets}(\mathcal{X})$ to refer to a multiset drawn from the domain of all multisets with elements from domain $\mathcal{X}$. We use the notation $\mathrm{MultiSet}(v)$ to refer to the multiset interpretation (unordered list with multiplicities) of vector v.

Remark 2.5. The distinction between multisets and vectors is relevant because vectors are ordered objects whereas multisets are not. In the OpenDP library, all datasets are represented as vector domains, and therefore for any vector v we need to use the notation $\mathrm{MultiSet}(v)$ when referring to its multiset representation to indicate that ordering should be dropped.

Example 2.6. For $\mathrm{MultiSet}(2, 3, 3, 5, 5, 5)$, element 2 has multiplicity 1, element 3 has multiplicity 2, and element 5 has multiplicity 3.

Remark 2.7. We remark that there is a bijection between multisets and histograms. Therefore, a proof can be carried out with either representation, although usually one will be simpler.

Definition 2.8 (Histogram notation, generic version)**.** Let $h_x : \mathcal{X} \rightarrow \mathbb{N}$ be the histogram of an (ordered or unordered) list x, where every element $\ell \in x$ is drawn from domain $\mathcal{X}$. We represent a histogram as a vector, where index i of histogram h_x, denoted $h_x(i)$, is equal to the number of occurrences of value i in list x. Therefore, we use $h_x(z)$ to denote the number of occurrences of every $z \in \mathcal{X}$ in list x (with multiplicities).

[2] Library most likely used in OpenDP to deal with floating point arithmetic with certain rounding modes.

3 Transformations & Stability relations

3.1 Transformations

Definition 3.1 (Transformation)**.** A transformation T is a *deterministic* mapping from arbitrary data types of derived data values to arbitrary data types of derived data values. In Rust, a transformation is specified by the following attributes: input domain, output domain, function, input metric, output metric, and stability relation.

See the pseudocode definitions document (see section 0.2) for further details on the pseudocode specification of a transformation.

3.2 Stability relations

Remark 3.2 (Purpose of stability relations)**.** Stability relations are used to tell the correctness of an upper bound on the distance between outputs of a function, given the distance between inputs to a function.

Definition 3.3 (Stability parameter)**.** For some value of c, we say that a transformation T is *c-stable* if for all x, x' in the input domain $\mathcal{X}$, and for input metric $d_{\mathcal{X}}$ and output metric $d_{\mathcal{Y}}$,

$$d_{\mathcal{Y}}(T(x), T(x')) \leq c \cdot d_{\mathcal{X}}(x, x'). \tag{1}$$

We say that c is the *stability parameter* of T.

Remark 3.4 (Lipschitz constant)**.** Note that the c-stable transformation T in definition 3.3 is also called a Lipschitz function with Lipschitz constant c. The stability parameter is analogous to the Lipschitz constant. The stability relation is a more robust relationship compared to the Lipschitz function and constant defined in other programming frameworks such as Fuzzi and PinQ.

There are two advantages for the notion of a stability relation over the linear Lipschitz function.

First, relation can capture a non-linear inequality, such as $\text{relation}(d_{in}, d_{out}) = (d_{out} \leq d_{in}^2)$. Second, the relation can capture the composition of more numbers for d_{in} or d_{out} respectively. We can define d_{in} as multiple privacy loss parameters in a composition.

In the OpenDP library, the stability parameter c (which in the case of clamping is equal to 1) gets wrapped up inside of the stability relation property, and the end user can test it empirically.

Definition 3.5 (Linear stability relation)**.** The *linear stability relation*, denoted by the term $\text{relation}(d_{in}, d_{out}) : \mathcal{X} \times \mathcal{Y} \to \textit{bool}$, is a boolean function which takes as input some d_{in}, d_{out} appropriately quantified and returns `True` if and only if the relation $d_{out} \geq c \cdot d_{in}$ for some specified value of c.

We can also write the stability relation in the following form:

$$\text{relation}(d_{in}, d_{out}) = \begin{cases} \texttt{True} & \text{if } d_{out} \geq c \cdot d_{in} \\ \texttt{False} & \text{otherwise,} \end{cases} \tag{2}$$

specifying the concrete metrics $d_{\mathcal{X}}, d_{\mathcal{Y}}$ and types of d_{in}, d_{out} inside the function.

Definition 3.6 (Generalized stability relation)**.** A *generalized stability relation*, also denoted $\text{relation}(d_{in}, d_{out}) : \mathcal{X} \to \mathcal{Y}$, is a boolean function which takes as input some appropriately quantified d_{in}, d_{out} and returns the result of the relation defined by $\text{relation}(d_{in}, d_{out})$. Unlike with the linear stability relation (defined in definition 3.5), the generalized stability relation can be any boolean function on d_{in}, d_{out}; it is not limited to being an inequality where d_{in}, d_{out} can only be related by a multiplicative constant c.

Example 3.7 (Generalized stability relation)**.** The stability relation for the "scalar clamp" function (no need to understand the "scalar clamp" function for the purposes of this example) is $d_{out} \geq \min(d_{in}, U - L)$. Note that the relationship between d_{out} and d_{in} is not linear.

3.3 Stability for Row Transforms

The following lemma can be applied to row by row transformations. This simplification allows us to avoid case by case analysis of the entire vector by focusing on function on the single row itself.

Definition 3.8 (Pure Function)**.** A pure function is a function that has the following properties:

- The function return values are identical for identical arguments.
- The function application has no side effects.

Note: This definition is taken from the "Pure function" article on Wikipedia.

Question for reviewers: Does the above definition of a "pure function" (definition 3.8) capture the traditional meaning of a pure function, or should anything be added/-modified? (Review for Marco)

Definition 3.9 (Vec)**.** We say that vector v is drawn from domain $\text{Vec}(\mathcal{X})$ if and only if all elements $v_i \in v$ are from domain $\mathcal{X}$.

Definition 3.10 (Row transform)**.** Let $f : \text{Vec}(\mathcal{X}) \to \text{Vec}(\mathcal{Y})$ be a function on vectors, and let $v = [v_1, \dots, v_n]$ be a vector of n elements. We say that f is a row transform with respect to a function $g : \mathcal{X} \to \mathcal{Y}$ if and only if $f(v) = [g(v_1), \dots, g(v_n)]$.

Remark 3.11. Intuitively, a row transformation works like `map` function in CS. It is a higher-order function that applies a given function to each row (or element) of the function.

Example 3.12 (*isEqual* is a row transform)**.** Let *bool* be the domain of boolean values. For every vector $v = [v_1, \dots, v_n] \in \text{Vec}(\mathcal{X})$, $isEqual(v, val) : \text{Vec}(\mathcal{X}) \times \mathbb{R} \to \text{Vec}(bool)$ is defined as $isEqual(v, val) = [v_1 == val, \dots, v_n == val]$.

Because *isEqual* applies the function $g(input) = (input == val)$ to every element of the vector, *isEqual* is a row transform.

Lemma 3.13 (Symmetric Distance of Row Transform)**.** *Let $f : \text{Vec}(\mathcal{X}) \to \text{Vec}(\mathcal{Y})$ be a row transform with respect to $g : \mathcal{X} \to \mathcal{Y}$. For every pair of vectors $v, w \in \text{Vec}(\mathcal{X})$, we have*

$$d_{Sym}(f(v), f(w)) \leq d_{Sym}(v, w).$$

Proof. We use the histogram notation. Recall that $h_{f(v)}(y)$ is the number of occurrences of y in vector $f(v)$. Therefore, $h_{f(v)}(y)$ is equivalent to the sum of the number of occurrences of each $x \in g^{(-1)}(y)$ in vector v. Since $h_{f(v)}(y) = \sum_{x \in g^{-1}(y)} h_v(x)$, we have:

$$\begin{aligned} \left|h_{f(v)}(y) - h_{f(w)}(y)\right| &= \left|\sum_{x \in g^{-1}(y)} h_v(x) - h_w(x)\right| \\ &\leq \sum_{x \in g^{-1}(y)} |h_v(x) - h_w(x)|. \end{aligned}$$

The last inequality follows by the triangle inequality. To compute the symmetric distance between $f(v)$ and $f(w)$, we have to sum over all possible elements $y \in \mathcal{Y}$, and then apply the inequality from above:

$$\begin{aligned} d_{Sym}(f(v), f(w)) &= \sum_{y \in \mathcal{Y}} \left|h_{f(v)}(y) - h_{f(w)}(y)\right| \\ &\leq \sum_{y \in \mathcal{Y}} \sum_{x \in g^{-1}(y)} |h_v(x) - h_w(x)| \end{aligned}$$

Note that because the sets $g^{-1}(y)$ form a partition of the domain of g, we can sum over elements x in the domain of g:

$$\sum_{y \in \mathcal{Y}} \sum_{x \in g^{-1}(y)} |h_v(x) - h_w(x)| = \sum_{x \in \mathcal{X}} |h_v(x) - h_w(x)| = d_{Sym}(v, w).$$

Therefore we have

$$d_{Sym}(f(v), f(w)) \leq d_{Sym}(v, w),$$

as desired. □

Remark 3.14. If the `make_row_by_row` transformation is directly invoked in the code (as opposed to only using the *row transform* as a proof strategy), then we require that f is a pure function. One such example would be the Impute Constant transformation.

3.4 Notes, todos, questions

4 Metrics

Whenever a metric is defined, this document will contain its mathematical definition. In turn, the pseudocode definitions document (see section 0.2) will include the list of compatible domains, the list of associated types, and the definition of d-close under said metric.

Definition 4.1 (Associated type)**.** The associated type of any input metric is the type of the corresponding d_{in}. In turn, the associated type of any output metric is the type of the corresponding d_{out}.

Remark 4.2. Whenever a metric is defined, there is only an associated type, while there are no input/output type. In the OpenDP programming framework, we do not think of metrics as functions in the usual sense; instead, the associated type of distance is the type of the corresponding d_{in} or d_{out}.

4.1 Dataset metrics

4.1.1 Symmetric distance

Definition 4.3 (Symmetric difference)**.** The *symmetric difference* between any two vectors u, v, denoted by $\mathrm{MultiSet}(u)\Delta\mathrm{MultiSet}(v)$, corresponds to the multiset representation of elements which are in either u or v but not in their intersection. The multiplicity of each element x in $\mathrm{MultiSet}(u)\Delta\mathrm{MultiSet}(v)$ corresponds to the difference in absolute value of the multiplicities of x in $\mathrm{MultiSet}(u)$ and in $\mathrm{MultiSet}(v)$.

Example 4.4. Because a multiset can have repeated elements, for $a = \mathrm{MultiSet}(1, 2, 1)$ and $b = \mathrm{MultiSet}(1, 3)$, we have $a\Delta b = \mathrm{MultiSet}(1, 3, 2)$.

We introduce the notion of symmetric distance, which differs from symmetric difference in that it is the *cardinality* of the multiset instead of the multiset itself.

Remark 4.5. Symmetric *distance* is the metric which is used in the OpenDP library, while the definition for symmetric *difference* is only included for completeness in this document.

Remark 4.6 (Inspiration for symmetric distance)**.** The notion of *symmetric distance* is inspired by the idea of calculating the cardinality of the symmetric difference between multisets. That is, the symmetric distance between any two (ordered or unordered) lists u, v, denoted $d_{Sym}(u, v) = |\mathrm{MultiSet}(u)\Delta\mathrm{MultiSet}(v)|$, can be thought of as the cardinality of the symmetric difference between the multiset interpretations of lists u and v.

Because there is a bijection between histograms and multisets, the symmetric distance between vectors u and v can also be thought of as the $L1$ distance between the histograms for u and v, denoted h_u and h_v (see definition 2.8). Then, we equivalently obtain that the symmetric distance between u and v is

$$d_{\mathrm{Sym}}(u, v) = d_{L1}(h_u, h_v) = \sum_{z\in\mathcal{X}} |h_u(z) - h_v(z)|.$$

The second equality follows from the definition of $L1$ distance (see definition 4.15).

We now proceed with the definition of symmetric distance, which takes these inspirations and generalizes them.

Definition 4.7 (Symmetric distance)**.** The symmetric distance between two (unordered or ordered) lists u, v, with every element in these lists drawn from domain $\mathcal{X}$, is

$$d_{Sym}(u, v) = \sum_{z\in\mathcal{X}} |h_u(z) - h_v(z)|. \tag{3}$$

Claim 4.8. Symmetric distance is a metric.

Note that null data values are still counted in the symmetric distance. Adding or removing null values still influences the count.

4.1.2 Substitute distance

Definition 4.9 (Substitute distance)**.** We only define the *substitute distance* d_{Sym} on multisets with the same number of elements. On two multisets $u, v \in \mathrm{MultiSet}(\mathcal{X})$ for some domain $\mathcal{X}$ where $|u| = |v|$, we say that the substitute distance $d_{Subs}(u, v)$ is equal to the cardinality of the relative complement $u \backslash v$, so $d_{Subs}(u, v) = |u \backslash v|$. (This can be thought of as fixing multiset u and finding how many elements in v are not represented in v.)

Alternatively, we can define d_{Subs} as

$$d_{Subs}(u, v) = \frac{1}{2} d_{\mathrm{Sym}}(u, v).$$

Remark 4.10 (Inspiration for substitute distance)**.** Note that substitute distance is like a generalization of Hamming distance to multisets (Hamming distance is defined only on ordered objects).

Question for reviewers: Should substitute distance (defined in definition 4.1.2) be able to be calculated on datasets that are not the same length (since it is defined as half of the symmetric distance, and since symmetric distance can be calculated on vectors of different length, too)? Or should substitute distance be limited to being calculated on vectors of the same length, since it is inspired by Hamming distance, which can only be calculated on vectors of the same length?

Claim 4.11. Substitute distance is a metric.

Remark 4.12. As of June 24, symmetric distance is a preferred metric over substitute distance. There is a constructor that converts the metric.

4.2 Sensitivity metric

4.2.1 Absolute distance

Definition 4.13 (Absolute distance)**.** Given two numbers n, m, the absolute distance between n and m, denoted by d_{Abs}, is defined as $d_{Abs}(n, m) = |n - m|$, where the vertical bars denote absolute value.

Claim 4.14. Absolute distance is a metric.

4.2.2 L1 distance

Definition 4.15 ($L1$ distance)**.** The *L1 distance* between any two vectors u, v, denoted by $d_{L1}(u, v)$, is defined as $d_{L1}(u, v) = \sum_{i=0}^{n} |u_i - v_i|$.

4.2.3 L2 distance

Definition 4.16 ($L2$ distance)**.** The *L2 distance* between any two vectors u, v, denoted by $d_{L2}(u, v)$, is defined as $d_{L2}(u, v) = \sqrt{\sum_{i=0}^{n} |u_i - v_i|^2}$.

4.2.4 Lp distance

More generally, we can define the L_p distance between two vectors

Definition 4.17 ($L2$ distance)**.** The *Lp distance* between any two vectors u, v, denoted by $d_{Lp}(u, v)$, is defined as $d_{Lp}(u, v) = (\sum_{i=0}^{n} |u_i - v_i|^p)^{1/p}$.

4.3 Closeness

Definition 4.18 (k-close)**.** For any metric d, we say that two elements u, v are k-close under d if $d(u, v) \leq k$.

For example, in the case of symmetric distance, vectors u and v are k-close whenever $d_{Sym}(u, v) = |\text{MultiSets}(v) \Delta \text{MultiSets}(u)| \leq k$. We remark that the type of k must correspond to the associated type of d.

We remark that the notion of k-closeness can be defined more generally without relying on metrics and instead only using d_{in}, d_{out} (e.g., (ϵ, δ)-DP). If that is the case, it will be defined accordingly.

4.4 Notes, todos, questions

The definitions in sections 5+ are not used in the most finalized proof documents. Please leave feedback, but note that the correctness of the following sections does not impact the correctness of the proofs that are most finalized.

5 Useful lemmas for simplifying proofs

5.1 Stability for Randomness

The following lemma and corollary are used to prove the stability guarantee holds in random transformations like `make_impute`.

Definition 5.1 (Coupling)**.** Let R and R' be two random variables defined over the probability spaces S and S', respectively. A *coupling* of R and R' is a joint variable (r, r') taking values in the product space $S \times S'$ such that r has the same marginal distribution as R and r' has the same marginal distribution as R'.

(grace) Taken from Jayshree's paper on Theil-Sen estimator.

Lemma 5.2 (Stability for Randomness)**.** *We define a randomized function* $f : \texttt{DI} \rightsquigarrow \texttt{DO}$*.* relation$(d_{in}, d_{out}) = True$ *implies that for all* $x, x' \in \texttt{DI}$ *that are* d_{in}*-close, there exists a coupling* (R, R') *of the randomness of* $f(x)$ *and* $f(x')$ *such that for all* $(r, r') \in Support(R, R')$*,* $f_r(x)$ *is* d_{out}*-close to* $f_{r'}(x')$*.*

Proof. (grace) Proof outline:

- Jayshree's paper shows how to do it for neighboring inputs x, x', then T is c-Lipschitz randomized transformation. More precisely, she says that if $dist(d_0, d_1) = 1$, then $dist(T(d_0), T(d_1)) \leq c$.
- We can generalize for any neighboring inputs: $d_Y(T(d_0), T(d_1)) \leq c \times d_X(d_0, d_1)$.
- Use the fact that c-stable implies Relation$(d_{in}, d_{out}) = d_{out} \geq c \times d_{in}$, $d_X(x, x') \leq d_{in}$, $d_Y(T(x), T(x')) \leq d_{out}$.

- Then apply Lemma 2.0.8 Jayshree's paper. Important things to note are that we find the probability distribution, which then shows that there exists a valid coupling of s_{d_0}, s_{d_l}. This allows us to conclude by triangle inequality distance between the coupled transformations $\mathrm{dist}(\tilde{s}_{d_0},\ \tilde{s}_{d_l}) \leq c \times l$. Therefore then T is a c-Lipschitz randomized transformation.
- If T is a c-Lipschitz randomized transformation, that means that $f_r(x)$ is d_{out}-close to $f_{r'}(x')$. (grace) Need to flesh out these details.

The following corollary allows us to fix the random seed in random transformations to prove the stability guarantee. □

Corollary 5.3. *For randomized function f : $\mathtt{DI} \rightsquigarrow \mathtt{DO}$,* relation$(d_{in}, d_{out})$ = $\mathtt{True}$ *implies that for all $x, x' \in \mathtt{DI}$ that are d_{in}-close and for all fixing of the randomness r of f (fixing seed of PRG), we have that $f_r(x)$ and $f_r(x')$ are d_{out}-close.*

5.2 The path property of symmetric distance on sized domains

Question: Is it a good approach to be avoiding texttt notation entirely in this section (and document)? E.g., then we need to define `SizedDomain` accordingly.

Definition 5.4 (*SizedDomain*)**.** *SizedDomain*$(\mathcal{X}, n)$ is the set of all vectors with n (not necessarily unique) elements drawn from domain $\mathcal{X}$.

Lemma 5.5 (Path property of d_{Sym} on *SizedDomain*)**.** *For any two vectors $v, w \in$ SizedDomain(D, n) for some domain D and integer n, $d_{Sym}(v, w)$ is an even integer; i.e., $d_{Sym}(v, w) = 2k$ for some integer $k \geq 0$. Moreover, there exist k vectors $v = u^0, u^1, \ldots, u^k = w$ such that $d_{Sym}(u^i, u^{i+1}) = 2$ for all i.*

Proof. We will first show that $d_{Sym}(v, w) = 2k$ for some integer $k \geq 0$. Since $v, w \in$ *SizedDomain(D, n)*, both v and w have length n. Let $v[i], w[i]$ denote the i-th element of v and w, respectively, for $i \geq 0$. We construct v', w' with the following iterative algorithm `Alg1`:

1. Set $v' \leftarrow v$ and $w' \leftarrow w$.
2. For each index $i \in [0, n-1]$, do:
    (a) (Deleting step.) Delete elements $v'[i]$ and $w'[j]$ if and only if there exists a $j \in [0, n-1]$ such that $v'[i] = w'[j]$.
3. Return v' and w'.

When `Alg1` terminates, it must be that $\mathrm{MultiSet}(v') \cap \mathrm{MultiSet}(w') = \emptyset$; otherwise, there would exist an index i such that $v'[i] = w'[j]$ for some j, which contradicts step 2 (a) in `Alg1`. Moreover, at each deleting step of the iterative algorithm, we delete exactly one element of v' and one element of w'. Let k' denote the total number of deletion steps. Since by assumption $|\mathrm{MultiSet}(v)| = |\mathrm{MultiSet}(w)| = n$, it follows that

$$|\mathrm{MultiSet}(v')| = |\mathrm{MultiSet}(w')| = n - k'.$$

Hence, $d_{Sym}(v, w) = 2k$, where $k = n - k'$.

Next we prove the second part of the lemma by using the same iterative algorithm. We show by construction how to find the k vectors $v = u^0, u^1, \dots, u^k = w$ such that $d_{Sym}(u^i, u^{i+1}) = 2 \forall\, i$. Let L denote the vector of indices j (in increasing order) such that $v[j]$ has not been deleted at any stage during the iterative algorithm `Alg1`. As shown above, we know that $|L| = k$. Let R denote the vector constructed with the elements of w that have not been deleted at any stage during `Alg1`, placed in the same ordering as in w. For the same reasons, $|R| = k$.

We now construct the k vectors u^i with the following algorithm `Alg2`:

1. Set $u^0 \leftarrow v$.

2. Set j = 0.

3. For each $i \in [1, k]$, do:

   (a) Set $u^i \leftarrow u^{i-1}$.

   (b) Set $u^i[L[i]] \leftarrow R[j]$.

   (c) Set $j \leftarrow j + 1$.

□

We need to prove two claims: that at the end of `Alg2`, $u^k = w$, and that $d_{Sym}(u^i, u^{i+1}) = 2$ for all i.

At each loop iteration of `Alg2`, only one element of vector u^i changes with respect to u^{i-1}, and all other elements are equal. Therefore, $d_{Sym}(u^i, u^{i+1}) = 2$. At the end of the k iteration, vector u^k consists of the $n - k$ original elements that v and w shared in the intersection (counting multiplicities), while the other k elements have been set equal by step 3 (b) in `Alg2`. Therefore, $u^k = w$, as we wanted to show.

Lemma 5.6. *For any two vectors $v, w \in$ SizedDomain(D, n), $d_{Sym}(v, w) = 2$ if and only if we can change one element of v to obtain v' such that* $\mathrm{MultiSet}(v') = \mathrm{MultiSet}(w)$.

Proof. This lemma follows directly form the definition of symmetric distance. We need to prove the two directions of the implication:

1. If we can change one element of v to obtain v' such that $\mathrm{MultiSet}(v') = \mathrm{MultiSet}(w)$, then $d_{Sym}(v, w) = 2$: This direction is already shown in the proof of Lemma 4.1.

2. If $d_{Sym}(v, w) = 2$, then we can change one element of v to obtain v' such that $\mathrm{MultiSet}(v') = \mathrm{MultiSet}(w)$: By the histogram notation, this means that $d_{Sym}(v, w) = \sum_{z \in \mathcal{X}} |h_v(z) - h_w(z)| = 2$. Hence, it must be that there exist two z_1 and z_2 such that $|h_v(z_1) - h_w(z_1)| = 1$ and $|h_v(z_2) - h_w(z_2)| = 1$, while all other terms in the sum are 0. Assume wlog that $z_1 \in v$ and $z_2 \in w$. Then, we can change z_1 to z_2 and we obtain a vector equal to w, as we wanted to see.

□

The above lemma follows directly from the definition of symmetric distance.

Lemma 5.7. *Given a function $f : \mathcal{X} \to \mathcal{Y}$ on input domain SizedDomain(D, n), if $d_{Sym}(f(v), f(w)) \leq c$ for all vectors $v, w \in$ SizedDomain(D, n) such that $d_{Sym}(v, w) = 2$, then f is $c/2$-stable.*

By the definition of c-stable (Definition 3.3), that f is $c/2$-stable is equivalent to stating that for all pairs s, t in the input domain, the following holds:

$$d_{Sym}(f(s), f(t)) \leq c/2 \cdot d_{Sym}(s, t).$$

Also equivalently, this means that the stability relation of f is

$$d_{out} \geq c/2 \cdot d_{in}.$$

Proof. Let s, t be two arbitrary vectors $\in SizedDomain(D, n)$ such that $d_{Sym}(s, t) \leq d_{in}$ for a fixed value of d_{in}. By Lemma 5.5, it follows that $d_{Sym}(s, t) = 2k$ for some integer $k \geq 0$, and that there exist k vectors $s = u^0, u^1, \ldots, u^k = t$ such that $d_{Sym}(u^i, u^{i+1}) = 2$ for all i. By the assumption in lemma 5.7, it follows that $d_{Sym}(f(u^i), f(u^{i-1})) \leq c$ for all i.

By the triangle inequality and lemma 5.6, it follows that

$$d_{Sym}(f(s), f(t)) \leq d_{Sym}(f(s), f(u^1)) + d_{Sym}(f(u^1), f(u^2)) + \ldots + d_{Sym}(f(u^{k-1}), f(u^k)) \leq$$

$$\leq kc.$$

Lastly, by Lemma 5.5, we know that $k = d_{Sym}(s, t)/2$, and hence by plugging it into k, we obtain that

$$d_{Sym}(f(s), f(t)) \leq c/2 \cdot d_{Sym}(s, t).$$

By definition, this means that f is $c/2$-stable, as we wanted to show. □

5.3 The path property: a generalization

Shortest path metric on a weighted graph with positive (non-zero) edge weights.

Lemma 5.8 (Generalized path property – generalization of lemma 5.5)**.** *Given an arbitrary domain-metric pair (D, d_M), we say that (D, d_M) has the path property if there exists a mapping $f : D \rightarrow V$, where $G(V, E)$ denotes a graph with positive non-zero edge weights and associated shortest path metric d_G, such that $\forall v, w \in D$, $d_M(v, w) = d_G(f(v), f(w))$.*

Note: We can either say "(D, M) has the path property" or "a metric on a domain has the path property".

(silvia) For lemma below, define the equivalent of $d_{Sym} = 2$ as the least possible value of d_M or as the distance between two adjacent vertices in the associated graph? Because MultiSet is specific to d_{Sym}. Or change to $d_G(f(v), f(w)) = d_{M_{\min}}$?

Question: Next lemma should work without the path property but I would need a general notion of adjacency; e.g., "if and only if v, w are adjacent elements". Redundant with $d_{M_{\min}}$? How to generalize the definitions correctly?

Lemma 5.9 (Generalization of Lemma 5.6)**.** *For any domain-metric pair (D, d_M) with the path property and hence with an associated graph $G(V, E)$, let $d_{M_{\min}}$ be the minimum possible value of $d_M(v, w)$ over of all possible input pairs $v, w \in D$ for $v \neq w$. Then, $\forall v, w \in D$, $d_M(v, w) = d_{M_{\min}}$ if and only if $f(v), f(w)$ are adjacent vertices in G.*

We say that two elements $v, w \in D$ are *adjacent* if and only if $d_M(v, w) = d_{M_{\min}}$.

No need to require the path property in Lemma 5.9 above (and hence have a graph) if v, w adjacent → $f(v), f(w)$ adjacent.

(silvia) It is important to specify that $d_{M_{\min}}$ is minimal *over D*. E.g., adjacent datasets are at $d_{Sym} = 1$ for unbounded domain, but at $d_{Sym} = 2$ for a bounded domain.

For any domain-metric pair (D, d_M) with the path property and hence with an associated graph $G(V, E)$, and for any function $f : D \to \mathcal{Y}$, where domain $\mathcal{Y}$ has associated metric $d_{\mathcal{Y}}$, if $d_{\mathcal{Y}}(f(v), f(w)) \leq c$ for all $v, w \in D$ such that $f(v), f(w)$ are adjacent vertices in G, then f is $c/d_{M_{\min}}$-stable.

Lemma 5.10 (Generalization of Lemma 5.7)**.** *For any domain-metric pair (D, d_M) with the path property and hence with an associated graph $G(V, E)$, and for any function $f : D \to \mathcal{Y}$, where domain $\mathcal{Y}$ has associated metric $d_{\mathcal{Y}}$, if $d_{\mathcal{Y}}(f(v), f(w)) \leq c$ for all $v, w \in D$ such that $d_M(u, v) = d_{M_{\min}}$, then f is $c/d_{M_{\min}}$-stable.*

How to generalize $d_{Sym} = 2$? There are 3 possibilities ($d_{Sym} \to d_M$):

- $d_M(u, v) = d_{M_{\min}}$
- u, v are adjacent elements under metric d_M (does this make any sense?)
- $f(u), f(v)$ are adjacent vertices in G (depends on the specific construction?)

Lastly, we show how Section 5.2 on the path property for sized domains is indeed a particular case of the generalized path property. By Lemma 5.8, to show that the domain-metric pair $(\mathit{SizedDomain}, d_{Sym})$ has the path property we need to provide the associated mapping $f : D \to V$, where $G(V, E)$ denotes a graph with positive non-zero edge weights and associated shortest path metric d_G such that $\forall v, w \in D, d_{Sym}(v, w) = d_G(f(v), f(w))$.

The mapping is as follows: for each v, w such that $d_{Sym}(v, w) = 2$ (i.e., they are neighboring datasets), we draw an edge of weight 2 between nodes $f(v), f(w)$ in G.

6 Measures

6.1 Max divergence

Definition 6.1 (Max divergence [1])**.** The max divergence between two random variables Y and Z taking values from the same domain is defined to be:

$$D_\infty(Y||Z) = \max_{S \subseteq \mathrm{Supp}(Y)} \left[\ln \frac{\Pr[Y \in S]}{\Pr[Z \in S]} \right].$$

Lemma 6.2 ([1])**.** *A mechanism $\mathcal{M}$ is ϵ-differentially private if and only if on every two neighboring databases x and y, $D_\infty(\mathcal{M}(x)||\mathcal{M}(y)) \leq \epsilon$ and $D_\infty(\mathcal{M}(y)||\mathcal{M}(x)) \leq \epsilon$.*

6.2 Smoothed max divergence

Definition 6.3 (Smoothed max divergence [1])**.** The smoothed max divergence between Y and Z is defined to be:

$$D_\infty^\delta(Y||Z) = \max_{S \subseteq \mathrm{Supp}(Y) : \Pr[Y \in S] \geq \delta} \left[\ln \frac{\Pr[Y \in S] - \delta}{\Pr[Z \in S]} \right].$$

Lemma 6.4 ([1])**.** *A mechanism $\mathcal{M}$ is (ϵ, δ)-differentially private if and only if on every two neighboring databases x and y, $D_\infty^\delta(\mathcal{M}(x)||\mathcal{M}(y)) \leq \epsilon$ and $D_\infty^\delta(\mathcal{M}(y)||\mathcal{M}(x)) \leq \epsilon$.*

7 Probability distributions

7.1 Laplace distribution

Definition 7.1 (Laplace distribution [1])**.** The Laplace distribution (centered at 0) with scale b is the distribution with probability density function

$$\mathrm{Lap}(x|b) = \frac{1}{2b} \exp\left(\frac{-|x|}{b}\right).$$

The variance of this distribution is $\sigma^2 = 2b^2$. (silvia) Add cumulative def.? Probably not necessary. Discuss anything else we would like to add.

We write $\mathrm{Lap}(b)$ to denote the Laplace distribution with scale b.
TODO (future – not enough info yet): Add Geometric and Gaussian.

7.2 Notes, todos, questions

8 Floating point approach

For now, this is summarized in the Overleaf document `https://www.overleaf.com/project/611159043230572a988ee69c`.

References

[1] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.* 9.3–4 (2014): 211–407.