Differential Privacy using Quasi-Borel Spaces

Michikazu Hirata

March 17, 2025

Abstract

This entry formalizes differential privacy using quasi-Borel spaces. In general, differential privacy is discussed using measurable spaces. Sato and Katsumata showed that quasi-Borel spaces are also applied to formulate differential privacy [1]. We formalize basic definitions and properties of differential privacy using quasi-Borel spaces, and show two examples: randomized response and the naive report noisy max algorithm.

Contents

theory *DP-QBS*
  imports *Differential-Privacy.Differential-Privacy-Divergence*
    Differential-Privacy.Differential-Privacy-Standard
    S-Finite-Measure-Monad.Monad-QuasiBorel
begin

declare *qbs-morphism-imp-measurable*[*measurable-dest*]

1 Definitions

Details of differential privacy using quasi-Borel spaces are found at [1]

1.1 Divergence for Differential Privacy using QBS

definition *DP-qbs-divergence* :: *'a qbs-measure* ⇒ *'a qbs-measure* ⇒ *real* ⇒ *ereal* (*DP'-divergence*$_Q$) **where**
DP-qbs-divergence-qbs-l: *DP-divergence*$_Q$ *p q e* ≡ *DP-divergence* (*qbs-l p*) (*qbs-l q*) *e*

abbreviation *DP-qbs-inequality* (*DP'-inequality*$_Q$) **where**
DP-qbs-inequality p q ε δ ≡ *DP-divergence*$_Q$ *p q* ε ≤ *ereal* δ

lemmas *DP-qbs-divergence-def* = *DP-qbs-divergence-qbs-l*[*simplified DP-divergence-SUP*]

lemma *DP-qbs-divergence-nonneg*[*simp*]: *0* ≤ *DP-divergence*$_Q$ *p q e*
⟨*proof*⟩

lemma *DP-qbs-divergence-le-ereal-iff*:
DP-divergence$_Q$ *p q* ε ≤ *ereal* δ ⟷ (∀ *A* ∈ *sets* (*qbs-l p*). *measure* (*qbs-l p*) *A* − *exp* ε ∗ *measure* (*qbs-l q*) *A* ≤ δ)
⟨*proof*⟩

corollary *DP-qbs-divergence-le-ereal-dest*:
assumes *DP-divergence*$_Q$ *p q* ε ≤ *ereal* δ
shows *measure* (*qbs-l p*) *A* ≤ *exp* ε ∗ *measure* (*qbs-l q*) *A* + δ
⟨*proof*⟩

corollary *DP-qbs-divergence-le-erealI*:
assumes ⋀ *A*. *A* ∈ *sets* (*qbs-l p*) ⟹ *measure* (*qbs-l p*) *A* ≤ *exp* ε ∗ *measure* (*qbs-l q*) *A* + δ
shows *DP-divergence*$_Q$ *p q* ε ≤ *ereal* δ
⟨*proof*⟩

lemma *DP-qbs-divergence-zero*:
assumes *p* ∈ *monadP-qbs X*
and *q* ∈ *monadP-qbs X*
and *DP-inequality*$_Q$ *p q 0 0*
shows *p* = *q*
⟨*proof*⟩

lemma *DP-qbs-divergence-antimono*: *a* ≤ *b* ⟹ *DP-divergence*$_Q$ *p q b* ≤ *DP-divergence*$_Q$ *p q a*
⟨*proof*⟩

lemma *DP-qbs-divergence-refl*[*simp*]: *DP-divergence*$_Q$ *p p 0* = *0*
⟨*proof*⟩

lemma *DP-qbs-divergence-refl'*[*simp*]: *0* ≤ *e* ⟹ *DP-divergence*$_Q$ *p p e* = *0*
⟨*proof*⟩

lemma *DP-qbs-divergence-trans′*:
assumes *DP-inequality*$_Q$ *p q* ε δ
and *DP-inequality*$_Q$ *q l* ε' *0*
shows *DP-inequality*$_Q$ *p l* $(\varepsilon + \varepsilon')$ δ
⟨*proof*⟩

lemmas *DP-qbs-divergence-trans* = *DP-qbs-divergence-trans′*[**where** δ=*0*]

proposition *DP-qbs-divergence-compose*:
assumes [*qbs,measurable*]:*p* ∈ *monadP-qbs X q* ∈ *monadP-qbs X f* ∈ *X* $\to_Q$ *monadP-qbs Y g* ∈ *X* $\to_Q$ *monadP-qbs Y*
and *dp1*:*DP-divergence*$_Q$ *p q* ε ≤ *ereal* δ
and *dp2*:⋀*x*. *x* ∈ *qbs-space X* ⟹ *DP-divergence*$_Q$ (*f x*) (*g x*) ε' ≤ *ereal* δ'
and [*arith*]: *0* ≤ ε *0* ≤ ε'
shows *DP-divergence*$_Q$ (*p* ⤜ *f*) (*q* ⤜ *g*) $(\varepsilon + \varepsilon')$ ≤ *ereal* $(\delta + \delta')$
⟨*proof*⟩

corollary *DP-qbs-divergence-dataprocessing*:
assumes [*qbs*]:*p* ∈ *monadP-qbs X q* ∈ *monadP-qbs X f* ∈ *X* $\to_Q$ *monadP-qbs Y*
and *dp*: *DP-divergence*$_Q$ *p q* ε ≤ *ereal* δ
and [*arith*]:*0* ≤ ε
shows *DP-divergence*$_Q$ (*p* ⤜ *f*) (*q* ⤜ *f*) ε ≤ *ereal* δ
⟨*proof*⟩

lemma *DP-qbs-divergence-additive*:
assumes [*qbs*]:*p* ∈ *monadP-qbs X q* ∈ *monadP-qbs X p′* ∈ *monadP-qbs Y q′* ∈ *monadP-qbs Y*
and *div1*: *DP-divergence*$_Q$ *p q* ε ≤ *ereal* δ
and *div2*: *DP-divergence*$_Q$ *p′ q′* ε' ≤ *ereal* δ'
and [*arith*]:*0* ≤ ε *0* ≤ ε'
shows *DP-divergence*$_Q$ $(p \bigotimes_{Qmes} p')$ $(q \bigotimes_{Qmes} q')$ $(\varepsilon + \varepsilon')$ ≤ *ereal* $(\delta + \delta')$
⟨*proof*⟩

corollary *DP-qbs-divergence-strength*:
assumes [*qbs*]:*p* ∈ *monadP-qbs X q* ∈ *monadP-qbs X x* ∈ *qbs-space Y*
and *dp*: *DP-divergence*$_Q$ *p q* ε ≤ *ereal* δ
and [*simp*]:*0* ≤ ε
shows *DP-divergence*$_Q$ (*return-qbs Y x* $\bigotimes_{Qmes}$ *p*) (*return-qbs Y x* $\bigotimes_{Qmes}$ *q*) ε ≤ *ereal* δ
⟨*proof*⟩

1.2 Differential Privacy using QBS

definition *DP-qbs* (*differential′-privacy*$_Q$) **where**
DP-qbs-qbs-L:*differential-privacy*$_Q$ *M* ≡ *differential-privacy* (λ*x*. *qbs-l* (*M x*))

lemma *DP-qbs-def*:
differential-privacy$_Q$ *M adj* ε δ ⟷
(∀(*d1*, *d2*)∈*adj*. *DP-inequality*$_Q$ (*M d1*) (*M d2*) ε δ ∧ *DP-inequality*$_Q$ (*M d2*) (*M d1*) ε δ)
⟨*proof*⟩

lemma *DP-qbs-adj-sym*:
assumes *sym adj*
shows *differential-privacy*$_Q$ *M adj* ε δ ⟷ (∀ (*d1*,*d2*) ∈ *adj*. *DP-inequality*$_Q$ (*M d1*) (*M d2*) ε δ)
⟨*proof*⟩

lemma *pure-DP-qbs-comp*:
assumes *adj* ⊆ *qbs-space X* × *qbs-space X*
and *adj′* ⊆ *qbs-space X* × *qbs-space X*
and *differential-privacy*$_Q$ *M adj* ε *0*
and *differential-privacy*$_Q$ *M adj′* ε′ *0*
and *M* ∈ *X* →$_Q$ *monadP-qbs Y*
shows *differential-privacy*$_Q$ *M* (*adj O adj′*) (ε + ε′) *0*
⟨*proof*⟩

lemma *pure-DP-qbs-trans-k*:
assumes *adj* ⊆ *qbs-space X* × *qbs-space X*
and *differential-privacy*$_Q$ *M adj* ε *0*
and *M* ∈ *X* →$_Q$ *monadP-qbs Y*
shows *differential-privacy*$_Q$ *M* (*adj* ^^ *k*) (*k* ∗ ε) *0*
⟨*proof*⟩

proposition *DP-qbs-postprocessing*:
assumes ε ≥ *0*
and *differential-privacy*$_Q$ *M adj* ε δ
and [*qbs*,*measurable*]:*M* ∈ *X* →$_Q$ *monadP-qbs Y*
and [*qbs*,*measurable*]:*N* ∈ *Y* →$_Q$ *monadP-qbs Z*
and *adj* ⊆ *qbs-space X* × *qbs-space X*
shows *differential-privacy*$_Q$ (λ*x*. *M x* ⤜ *N*) *adj* ε δ
⟨*proof*⟩

corollary *DP-qbs-postprocessing-return*:
assumes ε ≥ *0*
and *differential-privacy*$_Q$ *M adj* ε δ
and *M* ∈ *X* →$_Q$ *monadP-qbs Y*
and *N* ∈ *Y* →$_Q$ *Z*
and *adj* ⊆ *qbs-space X* × *qbs-space X*
shows *differential-privacy*$_Q$ (λ*x*. *M x* ⤜ (λ*y*. *return-qbs Z* (*N y*))) *adj* ε δ
⟨*proof*⟩

lemma *DP-qbs-preprocessing*:
  assumes $\varepsilon \geq 0$
    and *differential-privacy*$_Q$ M *adj* ε δ
    and [*measurable*]:$f \in X' \rightarrow_Q X$
    and $\forall (x,y) \in adj'. ((f\ x), (f\ y)) \in adj$
    and *adj* $\subseteq$ *qbs-space* $X \times$ *qbs-space* X
    and *adj'* $\subseteq$ *qbs-space* $X' \times$ *qbs-space* X'
  shows *differential-privacy*$_Q$ $(M \circ f)$ *adj'* ε δ
  ⟨*proof*⟩

proposition *DP-qbs-bind-adaptive*:
  assumes $\varepsilon \geq 0$ **and** $\varepsilon' \geq 0$
    and [*qbs*]:$M \in X \rightarrow_Q$ *monadP-qbs* Y
    and *differential-privacy*$_Q$ M *adj* ε δ
    and [*qbs*]:$N \in X \Rightarrow_Q Y \Rightarrow_Q$ *monadP-qbs* Z
    and $\bigwedge y.\ y \in$ *qbs-space* $Y \Longrightarrow$ *differential-privacy*$_Q$ $(\lambda x.\ N\ x\ y)$ *adj* ε' δ'
    and *adj* $\subseteq$ *qbs-space* $X \times$ *qbs-space* X
  shows *differential-privacy*$_Q$ $(\lambda x.\ M\ x \gg= N\ x)$ *adj* $(\varepsilon + \varepsilon')$ $(\delta + \delta')$
⟨*proof*⟩

proposition *DP-qbs-bind-pair*:
  assumes $\varepsilon \geq 0$ $\varepsilon' \geq 0$
    and [*qbs*]:$M \in X \rightarrow_Q$ *monadP-qbs* Y
    and *differential-privacy*$_Q$ M *adj* ε δ
    and [*qbs*]:$N \in X \rightarrow_Q$ *monadP-qbs* Z
    and *differential-privacy*$_Q$ N *adj* ε' δ'
    and *adj* $\subseteq$ *qbs-space* $X \times$ *qbs-space* X
  shows *differential-privacy*$_Q$ $(\lambda x.\ M\ x \gg= (\lambda y.\ N\ x \gg= (\lambda z.$ *return-qbs* $(Y \bigotimes_Q Z)\ (y,z))))$ *adj* $(\varepsilon + \varepsilon')$ $(\delta + \delta')$
⟨*proof*⟩

end

2 Examples

theory *DP-QBS-Examples*
  imports *DP-QBS*
    Differential-Privacy.Differential-Privacy-Randomized-Response
begin

lemma *qbs-space-list-qbs-borel*[*qbs*]: $\bigwedge r.\ r \in$ *qbs-space* (*list-qbs borel*$_Q$)
  and *qbs-space-list-qbs-count-space*[*qbs*]: $\bigwedge i.\ r \in$ *qbs-space* (*list-qbs* (*count-space*$_Q$ (*UNIV* :: - :: *countable*)))
  ⟨*proof*⟩

2.1 Randomized Response

lemma *qbs-morphism-RR-mechanism*[*qbs*]: *qbs-pmf* $\circ$ *RR-mechanism e* $\in$ *count-space*$_Q$ *UNIV* $\to_Q$ *monadP-qbs* (*count-space*$_Q$ *UNIV*)
⟨*proof*⟩

lemma *qbs-DP-RR-mechanism*:
assumes [*arith*]:$\varepsilon \geq 0$
shows *DP-divergence*$_Q$ (*RR-mechanism* ε *x*) (*RR-mechanism* ε *y*) $\varepsilon = 0$
⟨*proof*⟩

2.2 Laplace Distribution in QBS

lemma *qbs-morphism-laplace-density*[*qbs*]:*laplace-density* $\in$ *borel*$_Q$ $\Rightarrow_Q$ *borel*$_Q$ $\Rightarrow_Q$ *borel*$_Q$ $\Rightarrow_Q$ *borel*$_Q$
⟨*proof*⟩

definition *qbs-Lap-mechanism* (*Lap′-mechanism*$_Q$) **where**
Lap-mechanism$_Q$ $\equiv$ λ*e x. if e* $\leq$ *0 then return-qbs borel*$_Q$ *x else density-qbs lborel*$_Q$ (*laplace-density e x*)

lemma *qbs-morphism-Lap-mechanism*[*qbs*]:*Lap-mechanism*$_Q$ $\in$ *borel*$_Q$ $\to_Q$ *borel*$_Q$ $\Rightarrow_Q$ *monadP-qbs borel*$_Q$
⟨*proof*⟩

lemma *qbs-l-Lap-mechanism*: *qbs-l* (*Lap-mechanism*$_Q$ *e r*) = *Lap-dist e r*
⟨*proof*⟩

lemma *qbs-Lap-mechanism-qbs-l-inverse*:*Lap-mechanism*$_Q$ *e x*= *qbs-l-inverse* (*Lap-dist e x*)
⟨*proof*⟩

proposition *qbs-DP-Lap-mechanism*:
assumes $\varepsilon > 0$ **and** $|x - y| \leq r$
shows *DP-divergence*$_Q$ (*Lap-mechanism*$_Q$ ($1 / \varepsilon$) *x*) (*Lap-mechanism*$_Q$ ($1 / \varepsilon$) *y*) ($r * \varepsilon$) = *0*
⟨*proof*⟩

2.3 Naive Report Noisy Max Mechanism

primrec *qbs-NaiveRNM* :: *real* $\Rightarrow$ *real list* $\Rightarrow$ *real qbs-measure* **where**
qbs-NaiveRNM ε [] = *return-qbs borel 0* |
qbs-NaiveRNM ε (*x* # *xs*) =
(*case xs of*
Nil $\Rightarrow$ *Lap-mechanism*$_Q$ ($1 / \varepsilon$) *x* |
y#*ys* $\Rightarrow$ *do* {*x1* $\leftarrow$ *Lap-mechanism*$_Q$ ($1 / \varepsilon$) *x*; *x2* $\leftarrow$ *qbs-NaiveRNM* ε *xs*;
return-qbs borel (*max x1 x2*)})

lemma *qbs-morhpism-NaiveRNM*[*qbs*]: *qbs-NaiveRNM* $\in$ *borel*$_Q$ $\Rightarrow_Q$ *list-qbs borel* $\Rightarrow_Q$ *monadP-qbs borel*$_Q$

⟨*proof*⟩

theorem *qbs-DP-NaiveRNM'*:
assumes *pos*[*arith*,*simp*]: $\varepsilon > 0$
and *length xs* = *n* **and** *length ys* = *n*
and *adj*: $(\sum i<n.\ |nth\ xs\ i - nth\ ys\ i|) \leq r$
shows $DP\text{-}divergence_Q\ (qbs\text{-}NaiveRNM\ \varepsilon\ xs)\ (qbs\text{-}NaiveRNM\ \varepsilon\ ys)\ (r * \varepsilon) = 0$
⟨*proof*⟩

definition *adj-naive-RNM* :: *real* ⇒ (*real list* × *real list*) *set* **where**
$adj\text{-}naive\text{-}RNM\ r \equiv \{(xs,ys).\ length\ xs = length\ ys \land (\sum i<length\ xs.\ |nth\ xs\ i - nth\ ys\ i|) \leq r\}$

theorem *qbs-DP-NaiveRNM*:
assumes *pos*: $\varepsilon > 0$
shows $differential\text{-}privacy_Q\ (qbs\text{-}NaiveRNM\ \varepsilon)\ (adj\text{-}naive\text{-}RNM\ r)\ (r * \varepsilon)\ 0$
⟨*proof*⟩

end

References

[1] T. Sato and S. Katsumata. Divergences on monads for relational program logics. *Mathematical Structures in Computer Science*, 33(45):427–485, 2023.