Author's response

We have addressed the technical corrections/clarifications listed in the Editor's decision from 17 March 2018 and submitted a revised version of the manuscript.

In the following we provide a point-by-point response to the Editor's comments and list the relevant changes introduced with our revision.

p. 3, l.26 : p. 2 l. 21: use "\citep" (bracket references) where needed. Check manuscript throughout

Done.

p. 11, l.29 : "But notice that, as discovered by Ionescu (2009) 30 in the context of vulnerability studies, measures that pick up the most (least) probable Val-value of a probability distribution do violate the monotonicity condition" -> If the Ionescu thesis available online? If not, you might need to give a bit more detail. I tried to find a counterexample (of most-probable Val violating monotonicity) and, at least in a few minutes, I couldn't find any. Is there any simple counterexample you could give?

The thesis is available online and we have provided the URL from which it can be downloaded in the references of our revision. A detailed explanation of why most/least probable measures violate monotonicity is given at pages 112-116 of Ionescu's thesis for the more general case in which meas is of type M X -> Y. We have referred to these pages in the revision of the manuscript. A counterexample for the case in which meas is of type M Val -> Val with Val representing natural numbers can be built as follows. Consider the probability distribution

ma = [(x, 0.3), (y, 0.3), (z, 0.4)]

with A = {x,y,z} and 0.3, 0.3 and 0.4 the probabilities of x, y and z, respectively. Let Val = Nat and f, g : A -> Val with

f x = 1   f y = 2   f z = 3

g x = 2   g y = 2   g z = 3

We have that f a $\leq$ g a for all a : A and

map f ma = [(1, 0.3), (2, 0.3), (3, 0.4)]

map g ma = [(2, 0.3), (2, 0.3), (3, 0.4)]

The most probable Val value of map f ma is 3 with probability 0.4. The most probable Val value of map g ma is 2 with probability 0.3 + 0.3 = 0.6. Thus, if we take meas to be the most probable value we have

meas (map f ma) = 3

meas (map g ma) = 2

in contradiction with measMon at page 27 of the revised manuscript. A similar counterexample can be constructed for the case in which meas takes the least probable event. Thus, most/least probable measures violate the monotonicity condition from page 27: it is possible to increase all elements of a probability distribution (possibly by zero increments) and having their measure to strictly decrease!

The counterexample is interesting for applications (taking the most probable event seems, at the first glance, a quite natural way of measuring uncertainties) but it does not add anything substantial to the study of the impacts of uncertainties on optimal emission policies. Thus, we have decided not to add it to our revision.

p. 4., l, 21: programmes machine checked to be correct. This seems to be a central notion, but the average reader will need to undrestand better in what sense the programme is checked to be 'correct'. What does 'correct' mean in this context?

We agree that an explanation is needed and have added a whole paragraph with a simple example (that of a program that computes the square root of a number) at page 4, lines 23-31 and at page 5 lines 1-10 of the revised manuscript.

p. 4, l 18 : specificy the meaning of the $\sqsubseteq$ symbol. I do not understand why in this case we can't simply use $\leq$ (see also p. 26 l. 13). Is this is a generalisaton of $\leq$ for arbitrary defined order relations?

No, there is no generalization but we have to distinguish between ($\leq$) : Nat -> Nat -> Bool and ($\sqsubseteq$) : Nat -> Nat -> Type: on the right hand side of BoundedBy n ms, All expects a value of type Nat -> Type. Here ($\sqsubseteq$) is just a short cut (that we hope readers will relate to ($\leq$)) for the standard Idris data type LTE. This is defined as follows:

```
data LTE : Nat -> Nat -> Type where
  LTEZero : LTE Z right
  LTESucc : LTE left right -> LTE (S left) (S right)
```

The data type LTE defines what it means for a natural number m to be smaller than another natural number n. This is in contrast with ($\leq$) : Nat -> Nat -> Bool that represents a computation that is, a decision procedure for LTE. The idea of LTE is that it defines a proof that m is smaller than n as a value that can be constructed in one of two disjoint ways. These correspond to the constructors LTEZero and LTESucc:

1) If m is zero it is just LTEZero with no arguments. This formalizes the idea that zero is smaller or equal to any natural number.
2) If m is the successor of a number m' and n is the successor of a number n' then a proof that m is smaller than n can only be constructed (by applying LTESucc) if one already has a proof that m' is smaller than n'.

The definition of LTE is inductive in much the same way as that of natural numbers in functional languages:

```
data Nat : Type where
  Zero : Nat
  Succ : Nat -> Nat
```

To make a concrete example: one can implement a proof that 2 is smaller than 42

```
prf : LTE 2 42
```

by applying LTESucc to a proof that 1 is smaller than 41

```
prf = LTESucc prf' where
  prf' : LTE 1 41
  prf' = LTESucc LTEZero
```

Here, we have applied LTESucc to a proof prf' that 1 is smaller than 41. In turn, prf' is constructed by applying once more LTESucc to a proof that 0 is smaller than 40. The latter is simply LTEZero.

We think that it would be too confusing to discuss these technicalities in the manuscript and the discussion would not bring any better understanding of the impacts of uncertainties on emission policies,

Such a technical level would probably also not be fair: at page 3 line 30, we have written that we do not assume our readership to be familiar with dependent types. Indeed, most of the work that we have done in preparing our original contribution was geared towards hiding the technicalities of the theory behind the application that we present.

Thus, in the revised manuscript, we have expanded a little bit on ($\sqsubseteq$) at lines 20-13 of page 4 but we have not introduced further technical details.