Delegating Computation on Costly Data

Pablo D. Azar[*] Silvio Micali[†]
MIT MIT

September 10, 2015

Abstract

Collecting and processing large amounts of data is becoming increasingly crucial in our society. We model this task as evaluating a function f over a large vector $x = (x_1, \ldots, x_n)$, which is unknown, but drawn from a publicly known distribution X. In our model learning each component of the input x is costly, but computing the output $f(x)$ has zero cost once x is known.

We consider the problem of a principal who wishes to delegate the evaluation of f to an agent, whose cost of learning any number of components of x is always lower than the corresponding cost of the principal.

We prove that, for every continuous function f and every $\varepsilon > 0$, the principal can—by learning a single component x_i of x—incentivize the agent to report the correct value $f(x)$ with accuracy ε.

[*]Email: pazar@mit.edu. Address: MIT Economics. 77 Massachusetts Ave. E19-750, Cambridge MA, 02139.
[†]Email: silvio@csail.mit.edu. Address: MIT CSAIL. 32 Vassar Street, 32-G644, Cambridge, MA 02139.

1 Introduction

Our society is more and more data intensive. Every day, firms need to gather and process multiple pieces of data to make products and decisions. In this paper we investigate how to delegate to a rational agent the process of first obtaining multiple pieces of data and then aggregate them into a "compact" final result. In our model, obtaining the pieces of data is expensive, but the algorithmic operations to process them are free.

A bit more formally, in our setting a risk-neutral principal wants to learn the output of a continuous function $f : \mathbb{R}^n \to \mathbb{R}$ on an input $x = (x_1, ..., x_n)$. The input x is unknown, but drawn from a known distribution X. Furthermore, the function f depends on all its inputs,[1] and once the entire input vector $(x_1, \dots, x_n)$ is known, $f(x_1, \dots, x_n)$ can be computed at no cost. Learning the inputs, however, is costly. Specifically, the principal can learn any k coordinates of x at a cost of $\gamma(k)$. There also exists a risk-neutral agent, who can learn any k coordinates of x at a smaller cost, $c(k) < \gamma(k)$, where both $c(k)$ and $\gamma(k)$ are increasing with k. Since the agent has a lower cost, it would be socially optimal for the agent to learn the whole vector x, and then compute $y = f(x)$ and report y to the principal, assuming that the principal's value from learning y is higher than the agent's cost $c(n)$. However, since the principal cannot monitor the agent's actions, she cannot necessarily trust him to learn the whole $(x_1, \dots, x_n)$ nor to report the correct y. Thus, we must find a way to incentivize the agent to act on the best interests of the principal in this new framework.

Let us illustrate the usefulness of our goals by means of two examples.

Example 1: Scientific Data. Each input x_i is the result of an expensive but replicable physical experiment, which the agent can perform more cheaply than the principal.

Example 2: Proprietary Data. An information company has a proprietary database about the behavior of n individuals, and the principal is an outsider wishing to obtain some aggregate function of these data in order—say—to price a new product. Here, the agent is the information company itself, and while the principal can recreate each individual's data from scratch with non-trivial cost, the information company —after sinking a fixed cost into creating its database—can retrieve each record very cheaply.

Although our framework and results apply also when the number of inputs $n > 1$ is small and each input component x_i consists of just a few digits, our results are most relevant when n is large and each x_i has a large number of significant digits, so that it would be impractical for an agent to report the entire input vector $(x_1, \dots, x_n)$ to the principal. Typically, in fact, it is when $(x_1, \dots, x_n)$ is huge that one wishes to deal instead with an aggregate value $f(x_1, \dots, x_n)$.[2] Indeed, when the input vector x is large,[3] the principal should not insist on the agent revealing all the data he has learned, but on his reporting the right answer $f(x)$!

[1] More formally, for all x and all $i \in \{1, ..., n\}$, there exists x'_i such that $f(x_{-i}, x_i) \neq f(x_{-i}, x'_i)$.

[2] As an example, when f is Lipschitz continuous with Lipschitz constant 1, so that $|f(x) - f(x')| < \delta$ when $\|x - x'\|_\infty < \delta$, the number of significant digits that we would need to send to communicate $f(x)$ increases with $\frac{1}{\delta}$, while the number of significant digits needed to commmunicate the entire vector $(x_1, \dots, x_n)$ increases at a rate that is n times larger.

[3] E.g., the Large Hadron Collider at CERN produces 30 petabytes of data every year [12].

We investigate delegation of computation over costly inputs in two settings, described below.

1.1 Exact Computation

In our first setting, the *exact computation case*, the principal wants to learn $y = f(x_1, \ldots, x_n)$ *exactly.* We show that, for an important class of functions, the principal can incentivize the agent to reveal $f(x)$ using a *direct mechanism.* Informally, in a direct mechanism

1. The agent
   a. chooses a subset S_A of the input coordinates and learns the correponding subvector $x_{S_A} = \{x_i : i \in S_A\}$ at a cost of $c(|S_A|)$, and
   b. reports a value $z(x_{S_A})$, allegedly equal to $y = f(x_1, \ldots, x_n)$.
2. The principal
   a. chooses a (random) subset S of the input coordinates and learns $x_S = \{x_i : i \in S\}$, and
   b. pays the agent a reward $R(z, x_S)$ which is a continuous function of z and x_S.

A direct mechanism is

(a) *incentive compatible* if the agent maximizes his expected payoff $\mathbb{E}_S[R(z, x_S) - c(|S_A|)]$ only by choosing $S_A = \{1, \ldots, n\}$ and announcing $z = f(x_1, \ldots, x_n)$,

(b) *individually rational* if the expected reward is at least $c(n)$, the cost to a truthful agent of learning the whole input vector $(x_1, \ldots, x_n)$.

Our First Result: The Mechanism $\mathcal{M}_1$ A function $f : [0, 1]^n \to \mathbb{R}$ is separable if $f(x_1, \ldots, x_n) = \sum_{i=1}^n f_i(x_i)$. We show that if f is separable and each f_i is bounded, then the principal can correctly learn $f(x_1, \ldots, x_n)$ by means of a direct, incentive compatible, and individually rational mechanism $\mathcal{M}_1$ in which the principal queries *just one input* herself.

As we shall see, mechanism $\mathcal{M}_1$ is a crucial subroutine for our more general mechanisms.

Our Second Result: The Limits of Direct Contracts. Our first result raises the question of whether direct contracts can be used to delegate the exact computation of functions that are not separable. In our second theorem, we show that this is not the case even for a very simple non-separable function. Indeed we show that for

$$f(x) = \sum_{i=1}^{n} \prod_{j \neq i} x_j$$

no direct contract can incentivize the agent to reveal $f(x)$ unless the principal queries the entire input vector $(x_1, \ldots, x_n)$ herself.

1.2 Approximate Computation

In our second setting, *the ϵ-approximate computation case*, the principal wants to learn z such that $|z - f(x)| \leq \epsilon$, for some arbitrarily small $\epsilon > 0$.

Our Third Result: Approximately Delegating Arbitrary Continuous Functions. We show that for any continuous function $f : [0,1]^n \to \mathbb{R}$, any input $x \in [0,1]^n$ and any $\epsilon > 0$, there exists a (non direct) mechanism $\mathcal{M}_2$ that incentivizes the agent to reveal a z which is within distance ϵ of $f(x)$.

In mechanism $\mathcal{M}_2$:

- The principal queries only one coordinate of the input vector x
- The principal and the agent interact in 2 rounds, in each of which one sends a message to the other.

Our Fourth Result: Round Optimality of $\mathcal{M}_2$ In our fourth result, we show that one-round mechanisms cannot be used for approximate delegation of arbitrary continuous functions unless the principal queries the whole input vector $(x_1, \ldots, x_n)$, which would defeat the purpose of delegating the computation in the first place. Accordingly, our mechanism $\mathcal{M}_2$ simultaneously minimizes the number of queries that the principal makes to the input vector, and the number of rounds of interaction between the principal and the agent.

1.3 Optimality and Unlimited liability

We note that our mechanisms $\mathcal{M}_1$ minimizes the number of queries and interaction rounds, among all incentive-compatible mechanisms that delegate separable functions, and $\mathcal{M}_2$ minimizes the number of queries and interaction rounds, among all incentive-compatible mechanisms that approximately delegate continuous functions. These properties hold for any environment where the agent and principal are risk neutral. In particular, they hold when the agent has limited liability and can only be paid a positive amount ex-post.

Of course, queries and rounds of interaction are not the only costs that the principal faces. She must also bear the monetary cost of paying the agent's reward. We highlight that there is one setting where we can simultaneously minimize all three of these costs: namely, when the agent also has unlimited liability.

With unlimited liability, the technique for minimizing the expected payment to the agent is well known [8]: the principal charges the agent a fixed participation fee, equal to the expected utility that the principal gets from the mechanism.[4] In this way, the principal can always find a mechanism that makes the agent's IR constraint bind, without affecting the number of queries or the number of communication rounds.

Auditing and Optimality When the agent has unlimited liability, there is a trivial incentive compatible, individually rational, and one-round mechanism in which the principal learns $f(x_1, \ldots, x_n)$ exactly, minimizes the expected reward, and makes (in expectation) arbitrarily few queries. Namely,

[4]The principal can compute this expected reward at zero cost because she knows the distribution X from which the true input x is drawn and can take expectations with respect to this distribution without gathering any costly information about the true input x. Of course, this would not be true in a model where the principal has to pay a cost for every algorithmic operation that she makes.

The ε-Auditing Mechanism

1. The agent reports a value z, allegedly equal to $y = f(x_1, \ldots, x_n)$
2. With probability $(1 - \varepsilon)$, the principal pays the agent 0.
3. With probability ε, the principal queries the entire input vector $(x_1, \ldots, x_n)$ and pays the agent $\frac{c(n)}{\varepsilon}$, if $f(x_1, \ldots, x_n) = z$, and 0 otherwise.

The ε-Auditing mechanism, however, is not meaningful in many cases: for instance, when $\gamma(k) = +\infty$ for some $k > 1$. Consider first our example 1, where each x_i is the result of a replicable scientific experiment. Assume $n = 100$, and each experiment takes one day for the agent to perform, but one year for the principal. Then, for every possible choice of ε, the agent should never agree to participate in the ε-Auditing Mechanism, as the principal will die before he could pay the agent a positive amount.

Consider now our example 2, where $(x_1, \ldots, x_n)$ is a proprietary dataset owned by an information company. If the principal does not have enough money (or time) to learn herself the entire vector $(x_1, \ldots, x_n)$, then the agent has no hope to be paid. (Problems also arise even when the agent is able to reveal and prove to the principal the value of each x_i in a very cheap manner. Indeed, the information company must reveal all of his proprietary data in order to get paid, and thus destroys its own business by enabling a competitor in the process.)

In any case, even when the ε-Auditing Mechanism can be meaningfully used, we must ask what is the 'optimal mechanism to use once the auditing state is reached'. This is the mechanism that would be observable (when there is no auditing, we only observe that there is no auditing). Thus, we will focus on mechanisms that always make at least one query to the input distribution. As we shall prove in sections 3 and 4, $\mathcal{M}_1$ and $\mathcal{M}_2$ both make only one query to the input distribution and, under unlimited liability, are therefore "optimal once the auditing state is reached" for their respective settings.

1.4 Additional Prior Work

In a prior setting, the buying and selling of information has been modeled via an agent, who (costlessly) knows a distribution X about an uncertain state of the world x, and a principal who wants to incentivize the agent to reveal the distribution X. This problem has been addressed via strictly proper scoring rules [4, 7]. We differ from this traditional setting in three important ways: (1) the agent has cost for gathering information, as in the work of Osband [15], Clemen [5] and Lambert [11], (2) the principal only wants to learn an aggregate function $f(x)$ of the uncertain state of the world, not the entire state itself, and (3) the principal can herself obtain any information that the agent can obtain, albeit at a higher cost. As we will show, however, we can adapt the techniques from scoring rules to apply to our different setting.

A different way to account for computation in the design of contracts has been studied by Anderlini and Felli [2], and Al-Najjar, Anderlini and Felli [1]. Anderlini and Felli show that when contracts are generated by a Turing Machine, then in some situations computable contracts will be sub-optimal. Al-Najjar Anderlini and Felli show that when events cannot be finitely described, sometimes the best contract is no contract at all. Since our model focuses on the computation of continuous functions, where the complexity of a contract depends only on the number of inputs queried, we bypass these impossibility results.

Another way in which complexity in contracts can be modeled is via how much time it takes to resolve uncertainty in the state of the world. Using this, MacLeod studies what type of contracts are more efficient in low and high complexity environments [13].

In a previous conference paper [3], we studied the problem of rational interactive proofs, where the inputs are costlessly available to both principal and agent, computation is free for the agent, and the principal is bound to act in polynomial time. Our results were in the setting of computational complexity, and we were able to show that such a principal can delegate combinatorial counting problems. In this setting we show characterizations for multiple counting complexity classes including $\#P$ and the counting hierarchy CH.

2 The Model

Notation We denote the set $\{1, \dots, n\}$ by $[n]$. For any $S \subset [n]$, $-S$ is the complement of S (i.e., $-S \overset{\text{def}}{=} [n] \setminus S$), and x_s is the function mapping each $i \in S$ to x_i. (We emphasize that knowledge of x_S implies knowledge of the underlying coordinate set S.) We refer to each subvector x_S as a partial input.

If $X = (X_1, \dots, X_n)$ is a random variable over $\mathbb{R}^n$, we define $X_S \overset{\text{def}}{=} (X_i)_{i \in S}$ and denote the conditional random variable "X given that $X_S = x_S$" by $X|x_S$. We refer to the process of learning x_S as *querying* the subset S.

If g is a function of a random variable x drawn from distribution X, we will write the expectation of g as $\mathop{\mathbb{E}}_{x \leftarrow X}[g(x)]$. We write the expectation of a function $g(x, y)$ with respect to two independent random variables x and y as $\mathop{\mathbb{E}}_{x \leftarrow X, y \leftarrow Y}[g(x, y)]$.

We will often work with vectors that grow in size. Given a vector $x = (x_1, \dots, x_n) \in \mathbb{R}^n$ and a real number y, we will denote the expanded vector $(x_1, \dots, x_n, y)$ by $x||y$. We will refer to a vector of length 0 as the empty vector.

Computational Environments A *computational environment* is a tuple, $\mathcal{E} = (f, X, x, c, \gamma)$, where

- $f : [0, 1]^n \to \mathbb{R}$ is the *target function.*
- $X \in \Delta([0, 1]^n)$ is a continuous distribution, which is common knowledge to the principal and the agent.
- $x = (x_1, \dots, x_n)$ is a random variable, a priori unknown to the principal or agent, drawn from X.
- $c : \{0, \dots, n\} \to \mathbb{R}$ is the *agent's cost function.* For every set $S \subset [n]$, $c(|S|)$ is the cost to the agent of learning x_S.

- $\gamma : \{0, \dots, n\} \to \mathbb{R}$, the *principal's cost function.* For every set $S \subset [n]$, $\gamma(|S|)$ is the cost to the principal of learning x_S.

Assumptions. Throughout the paper, we will have the following assumptions

0 Any sequence of purely algorithmic operations has zero cost for both the principal and agent. (Thus, evaluating any function on known inputs, or computing an expectation over a known distribution, can be performed at zero cost.)

1. For all proper subsets $S \subsetneq [n]$, and all realizations x_S of X_S, the conditional random variable $f(X)|x_S$ has a continuous probability distribution which assigns probability zero to any individual point in the range of f.

2. The cost functions are monotonic. For all $\ell < k$ we have
$$\gamma(\ell) \leq \gamma(k), c(\ell) \leq c(k').$$

3. The cost of querying zero inputs is zero
$$\gamma(0) = c(0) = 0.$$

4. The cost functions satisfy the increasing differences condition. For all ℓ, k we have
$$c(k + \ell) - c(k) \leq \gamma(k + \ell) - \gamma(k).$$
(I.e., it is always cheaper for the agent than for the principal to query ℓ extra inputs.)

3 Exact Computation

In this section, we prove our results for the exact computation setting. Before stating our results, we define direct computational contracts, define boundedly separable functions, and recall some facts about proper scoring rules.

3.1 Preliminaries

Definition 1.
For all $k \leq n$, a ***k-query direct computational contract*** *is a mechanism $\mathcal{M}$ specified by*

- *a function $\mathcal{D}$ mapping a real number z to a distribution $\mathcal{D}(z)$ over $\{S \subset [n] : |S| = k\}$, that is, over all input coordinate subsets of size k*
- *a continuous and concave reward function R mapping a real number z, and a partial input x_S to a real number $R(z, x_S)$.*

Such a mechanism $\mathcal{M} = (\mathcal{D}, R)$ has only one player, the agent, and is played as follows
Stage 0. Nature draws $x \leftarrow X$. (The agent does not observe x.)

Stage 1. The agent queries a subset S_A of the inputs (updating his beliefs about x to $X|x_{S_A}$).
Stage 2. The agent reports a value $z = z(x_{S_A})$ to the mechanism.
Stage 3. The mechanism draws a subset S of inputs from the distribution $\mathcal{D}(z)$.

In such a play, the agent's reward is $r = R(z, x_S)$ and his utility is $r - c(|S_A|)$.

*$\mathcal{M}$ is **incentive compatible** for a function $f : [0,1]^n \to \mathbb{R}$ if the agent strictly maximizes his utility by choosing $S_A = [n]$ in stage 1, and $z = f(x)$ in stage 2.*

*$\mathcal{M}$ is **individually rational** for f if* $\underset{S \leftarrow \mathcal{D}, x \leftarrow X}{\mathbb{E}}[R(f(x), x_S) - c(n)] \geq 0$

Definition 2. *A function $f : [0,1]^n \to \mathbb{R}$ is **boundedly separable** if there exist bounded functions $f_1, \ldots, f_n : [0,1] \to \mathbb{R}$ such that $f(x_1, \ldots, x_n) = \sum_{i=1}^n f_i(x_i)$.*

Scoring Rules A *strictly proper scoring rule*[5] is a function $S : \Delta(K) \times K \to \mathbb{R}$ which takes as input a distribution Ω over a finite set K and a sample $\omega \in K$ and which satisfies the property

$$\underset{\omega \leftarrow \Omega}{\mathbb{E}} S(\Omega, \omega) > \underset{\omega \leftarrow \Omega}{\mathbb{E}} S(\Omega', \omega)$$

for any $\Omega' \neq \Omega$. A function which satisfies this property incentivizes a rational expert to state his true beliefs about the distribution of ω.

Many such rules are known by now.[6] We will use Brier's scoring rule, defined by

$$\mathtt{BSR}(\Omega, \omega) = 2Pr(\Omega = \omega) - \sum_{\alpha} Pr(\Omega = \alpha)^2 + 1$$

where $Pr(\Omega = \alpha)$ is the probability that distribution Ω assigns to the element α. The Brier Scoring Rule is always in the interval $[0, 3]$. [7]

3.2 Our First Theorem

Theorem 1. *Let $f : [0,1]^n \to \mathbb{R}$ be a boundedly separable function. Then there exists a one-query, incentive compatible and individually rational direct computational contract $\mathcal{M}_1 = (R, \mathcal{D})$ for f.*

Proof. We first prove theorem 1 assuming that the agent has zero costs: that is, $c(|S_A|) = 0$ for all $S_A \subset [n]$. Let $f(x_1, \ldots, x_n) = f_1(x_1) + \ldots + f_n(x_n)$ and let $B > 0$ be a bound such that $|f_i(x)| < B$ for every $x \in [0,1]$ and every $i \in [n]$. Let $g_i(x) = f_i(x_i) + B$ and note that

[5]For brevity and when the context is clear, we will often refer to strictly proper scoring rules simply as *proper scoring rules* or *scoring rules*.

[6]The interested reader is referred to a paper by Gneiting and Rafterty [6], which includes a comprehensive survey.

[7]Usually, the Brier Scoring Rule was defined as $\mathtt{BSR}(\Omega, \omega) = 2Pr(\Omega = \omega) - \sum_\alpha Pr(\Omega = \alpha)^2 - 1$. Our formula is the usual definition, plus 2. The reason we add 2 to the usual formulation of the scoring rule is to ensure that the reward to the expert is non-negative. Note that adding a constant to this reward does not affect incentives.

$0 \le g_i(x) \le 2B$. Let $g(x_1, \ldots, x_n) = \sum_{i=1}^n g_i(x) = f(x_1, \ldots, x_n) + nB$. We now give an incentive compatible contract for g. Since, by construction, we have ensured that each term $g_i(x)$ in the sum is non-negative, our mechanism can scale this term so as to interpret it as a probability, and use some techniques from scoring rules to incentivize the agent.

Our mechanism $\mathcal{M}_1$ takes the agent's report z as an input and produces a distribution $\mathcal{D}(z)$ and reward function $R(z, \cdot)$ as follows

MECHANISM $\mathcal{M}_1(z) = (\mathcal{D}(z), R(z, \cdot))$

- $\mathcal{D}(z)$ is the uniform distribution over the singleton sets $\{\{1\}, \ldots, \{n\}\}$.[a]
- The reward function $R(z, \cdot)$ is defined as follows
  - If $z \notin [-Bn, Bn]$, then $R_1(z, x_S) = -\infty$ for all x_S.
    (I.e., since the range of f is $[-Bn, Bn]$, a z outside this range must be a lie.)
  - If $z \in [-Bn, Bn]$, then the mechanism
    * Draws $S = \{i\}$ from $\mathcal{D}$. Since S is a singleton we denote it by $S = i$.
    * Queries x_i and computes $g_i(x_i)$.
    * Draws a random variable ω which is
      equal to 1 with probability $\frac{g_i(x_i)}{2B}$ and
      equal to 0 with probability $1 - \frac{g_i(x_i)}{2B}$.
    * Interprets z as a random variable Ω_z over $\{0, 1\}$ which is
      equal to 1 with probability $\frac{z+Bn}{2Bn}$ and
      equal to 0 with probability $1 - \frac{z+Bn}{2Bn}$.
    * Returns $R_1(z, x_i) = BSR(\Omega_z, \omega)$.

[a]Although the definition allows $\mathcal{D}$ to depend on z, in this case $\mathcal{D}$ does not depend on the agent's report. The same holds for our mechanism $\mathcal{M}_2$ for delegating any continuous function. The ability of $\mathcal{D}$ to depend on the agent's report is important to achieve the right level of generality, and thus gives meaningfulness to our Theorem 2, which argues that $n-1$-query direct contracts cannot be used to delegate non-separable functions.

Let us now show that mechanism $\mathcal{M}_1$ is incentive compatible. Given a report z, the agent's expected reward is $\mathbb{E}_i\, R(z, x_i) = \mathbb{E}_\omega\, BSR(\Omega_z, \omega)$. Since Brier's scoring rule is strictly proper, this expected reward is maximized only when the agent reports z so that Ω_z is equal to the distribution from which the principal is drawing ω.

Let us now consider the probability that ω is equal to 1. This probability is equal to $\sum_{i=1}^n Pr(\omega = 1 | S = i) Pr(S = i)$. Since $Pr(\omega = 1 | S = i) = \frac{g_i(x_i)}{2B}$ and $Pr(S = i) = \frac{1}{n}$, we have that

$$Pr(\omega = 1) = \sum_{i=1}^n Pr(\omega = 1 | S = i) Pr(S = i) = \frac{1}{2Bn} \sum_{i=1}^n g_i(x_i).$$

Since Ω_z is a random variable with $Pr(\Omega_z = 1) = \frac{z+Bn}{2Bn}$, the agent maximizes his reward by announcing z such that $z + Bn = \sum_i g_i(x_i)$. Note that this is equivalent to announcing $z = \sum_i f_i(x_i)$, because each $g_i(x_i) = f_i(x_i) + B$. Thus, the agent maximizes his reward by announcing $z = f(x_1, \ldots, x_n)$. Note that since we are currently assuming that the agent's cost function is identically 0, and since—by assumption 1—the agent cannot know $f(x_1, \ldots, x_n)$ exactly unless he queries all the inputs, this mechanism incentivizes him to query $S_A = [n]$.

The above argument shows that once the agent knows the entire vector $(x_1, \ldots, x_n)$, then he is incentivized via a scoring rule approach to report $z = f(x_1, \ldots, x_n)$. The problem is that this argument only applies when the agent's cost function c is identically 0. When this is not the case, we now prove that we can scale the reward $R(z, x)$ by a large enough constant so that the agent is incentivized to query the whole vector $(x_1, \ldots, x_n)$. As mentioned above, once the agent has learned $(x_1, \ldots, x_n)$, the scoring rule guarantees that he will maximize his reward by reporting $z = f(x_1, \ldots, x_n)$. This part of our proof is standard in the costly information acquisition literature [11, 15, 5] and proceeds as follows.

For every partial input x_{S_A}, let

$$z^*(x_{S_A}) \in \underset{z}{\operatorname{argmax}} \underset{i \leftarrow \mathcal{D}_1, x \leftarrow X | x_{S_A}}{\mathbb{E}} [R(z, x_i)]$$

$$\nu(S_A) = \underset{i \leftarrow \mathcal{D}_1, x \leftarrow X}{\mathbb{E}} [R(z^*(x_{S_A}), x_i)]$$

so that $\nu(S_A)$ is the reward that the agent ultimately expects to receive when he chooses to query set S_A. Since the agent maximizes his expected reward by announcing $z = f(x_1, \ldots, x_n)$, and since he can only learn $f(x_1, \ldots, x_n)$ by querying the whole set, we have that $\nu([n]) > \nu(S_A)$ for any $S_A \subsetneq [n]$.

When we account for the agent's cost, he is incentivized to query $S_A = [n]$ if and only if $[n] = \operatorname{argmax}_{S_A} \nu(S_A) - c(|S_A|)$. Let $\kappa > 0$ be such that, for any $S_A \subsetneq [n]$,

$$\kappa \cdot \nu(\{1, \ldots, n\}) - c(n) > \kappa \cdot \nu(S_A) - c(|S_A|). \tag{1}$$

Such a κ exists because $\nu([n]) > \nu(S_A)$ for any proper subset S_A. Furthermore, since $\nu(S_A)$ can be computed by taking expectations over the commonly known distribution X, and without the need to learn the true input x, the principal can compute κ at zero cost. Let $\widetilde{R}(\cdot, \cdot) \overset{\text{def}}{=} \kappa \cdot R(\cdot, \cdot)$ be a scaled reward function, and let $\widetilde{\nu}(S_A)$ be the reward that the agent expects to receive when he chooses to query S_A and the reward function is $\widetilde{R}$. Then, by construction, we have

$$\widetilde{\nu}([n]) - c(n) > \widetilde{\nu}(S_A) - c(|S_A|)$$

for any proper subset S_A. Thus, by changing the reward of function of $\mathcal{M}_1$ to be $\kappa \cdot R$, we can incentivize the agent to query the whole input $(x_1, \ldots, x_n)$ and to report $f(x_1, \ldots, x_n)$.

Finally, since $\nu([n])$ is positive, we can choose κ so that inequality (1) holds, and also $\kappa \cdot \nu(\{1, \ldots, n\}) - c(n) \geq 0$. That is, so that $\mathcal{M}_1$ using the reward function $\kappa \cdot R$ is also individually rational.

Q.E.D.

3.3 Query Optimality and Our Second Theorem

Since the mechanism $\mathcal{M}_1$ only makes one query we have the following corollary

Corollary 1. *No direct computational contract that is incentive-compatible for separable functions makes fewer queries than $\mathcal{M}_1$.*

We now argue that the query optimality of mechanism $\mathcal{M}_1$ is intrinsically linked to the fact that f is separable. Indeed, we show the following theorem

Theorem 2. *There exists a continuous, bounded and non-separable function $f : [0,1]^n \to \mathbb{R}$ such that any direct computational contract $\mathcal{M} = (\mathcal{D}, R)$ that is incentive compatible for f must query the whole input vector $(x_1, \ldots, x_n)$.*

The proof of Theorem 2 is given in appendix A.

4 Approximate Computation

Direct computational contracts are sufficient for delegating boundedly separable functions. But to handle the delegation of arbitrary continuous functions in an approximate manner, we need to define computational contract more generally, so as to allow multiple rounds of communication between principal and agent. In addition, before proving our theorems, we recall a result of Kolmogorov about representing continuous functions, as well as a basic fact about Brier's Scoring Rule.

4.1 Preliminaries

Definition 3.
A **k*-query,* T*-round computational contract*** *for a function $f : [0,1]^n \to \mathbb{R}$ is a mechanism $\mathcal{M}$ specified by*

- *A function $\mathcal{D}$ mapping a vector m to a distribution $\mathcal{D}(m)$ over a finite support D.*
- *A function $\mathcal{S}$ mapping a vector m to a set of input coordinates $\mathcal{S}(m)$ of size k,*
- *a continuous and concave reward function R mapping a vector m and a partial input x_S to a real number $R(m, x_S)$.*

Such a mechanism $\mathcal{M} = (\mathcal{D}, \mathcal{S}, R)$ has a single player, the agent, and it is played over T rounds. In each round, the agent sends a message to the mechanism and then receives a random message from the mechanism.

At any round $t \in \{1, \ldots, T\}$, the information available to the agent consists of the set S_A^t of all inputs he has queried so far, and of the vector m^t of all messages exchanged with the mechanism so far. Initially, $S_A^0 = \emptyset$ and m^0 is the empty vector. A play of the mechanism proceeds as follows.

Stage 0. Nature draws $x \leftarrow X$. The agent does not observe x.

Round t. For each $t \in \{1, \ldots, T\}$, round t consists of the following stages:

Stage $2(t-1)+1$. The agent
chooses a function $a_t(\cdot)$ and a subset $S_{A,t} \subset [n]$,
queries the set $S_{A,t}$ and updates $S_A^t = S_A^{t-1} \cup S_{A,t}$,
sends the mechanism the message $m_t = a_t(x_{S_A^t})$ and updates $m^t = m^{t-1} || m_t$.

Stage $2(t-1)+2$. The mechanism draws a random element r_t from the distribution $\mathcal{D}_t = \mathcal{D}(m^t)$, and the agent updates his set of messages to $m^t = m^t || r_t$.

At the end of this play, the mechanism queries the set $S = \mathcal{S}(m^T)$ and pays the agent the reward $r = R(m^T, x_S)$. The agent's utility is $r - c(|S_A^T|)$.

*$\mathcal{M}$ is ϵ-**incentive compatible** for f if there exists a function $h : \mathbb{R}^T \to \mathbb{R}$ such that*

$$|h(m_1, \ldots, m_T) - f(x)| < \epsilon$$

where the sequence of messages $m_1, \ldots, m_T$ is generated by an agent that, at each round $t \in \{1, \ldots, T\}$ chooses $a_t(\cdot)$ and $S_{A,t}$ to maximize his expected utility given the information $x_{S_A^{t-1}}, m^{t-1}$ available to him at the beginning of the round.

*$\mathcal{M}$ is **individually rational** if*

$$\max_{a_1(\cdot), S_{A,1}} \mathop{\mathbb{E}}_{x \leftarrow x, r_1 \leftarrow \mathcal{D}_1} \ldots \max_{a_T(\cdot), S_A^T} \mathop{\mathbb{E}}_{x \leftarrow X | (x_{S_A^{T-1}}, m^{T-1}), r_T \leftarrow \mathcal{D}_T} \left[R((m_1, r_1, \ldots, m_T, r_T), x_{\mathcal{S}(m)}) \right] - c(|S_A^T|) > 0$$

Kolmogorov's Superposition Theorem In Theorem 3, we will make use of the following representation of continuous functions over compact sets.

Let $f : [0,1]^n \to \mathbb{R}$ be an arbitrary continuous function. Then f has the representation

$$f(x) = \sum_{q=0}^{2n} \Phi(\sum_{p=1}^{n} \psi_{q,p}(x_p)) \tag{2}$$

where $\Phi_q, \psi_{q,p}$ are continuous one-dimensional functions and the functions $\psi_{q,p}$ are Lipschitz continuous and independent of the function f.

A Basic Property of Brier's Scoring Rule In our proof of Theorem 3, we will use the following well known property of Brier's scoring rule on binary distributions

LEMMA 1 *Let v, w be real numbers in $[0,1]$ and let V, W be random variables over $\{0,1\}$ such that $Pr(V=1) = v, Pr(W=1) = w$. Then*

$$\mathop{\mathbb{E}}_{\omega \leftarrow V}[BSR(V, \omega) - BSR(W, \omega)] = 2(v-w)^2$$

For completeness, the proof is given in Appendix B.

4.2 Our Third Theorem

Theorem 3. *For all continuous functions $f : [0,1]^n \to \mathbb{R}$ and all $\epsilon > 0$, there exists a 1-query, 2-round, computational contract $\mathcal{M}_2$ that is ϵ-incentive compatible and individually rational.*

Proof. As in the proof of Theorem 1, we first prove Theorem 3 assuming that the agent's cost function is identically zero. We then use this result to prove the more general result when the agent's cost function is arbitrary.

Case 1: $\mathbf{c}(\cdot) \equiv \mathbf{0}$. Let $f(x_1, \ldots, x_n) = \sum_{q=0}^{2n} \Phi(\sum_{p=1}^{n} \psi_{q,p}(x_p))$ as in Kolmogorov's Superposition Theorem. Since the functions $\psi_{q,p}$ are Lipschitz continuous, there exists an M such that $|\psi_{q,p}(x) - \psi_{q,p}(x')| < M|x - x'|$ for any $x, x' \in [0,1]$. Furthermore, because of this Lipschitz condition, the family of functions $\{\psi_{q,p}\}_{q,p}$ has the following two properties

- Uniform boundedness: there exists a constant $B > 0$ such that $|\psi_{q,p}(x)| \leq B$ for every $x \in [0,1]$ and every q, p.
- Uniform equicontinuity: for every $\epsilon > 0$ there exists a $\delta > 0$ such that $|x - x'| < \delta$ implies $|\psi_{q,p}(x) - \psi_{q,p}(x')| < \epsilon$ for every q, p and every $x, x' \in [0,1]$.

Because of uniform boundedness, we can interpret the domain of the "outer" function Φ as the compact set $[-nB, nB]$. Since any continuous function with a compact domain is bounded and uniformly continuous, we have that Φ is bounded and uniformly continuous. Denote by C the bound on Φ.

Intuition. The intuition behind our contract is to interpret $f(x) = \sum_{q=0}^{2n} \Phi(\sum_{p=1}^{n} \psi_{q,p}(x_p))$ as a boundedly separable function $\widetilde{f}(w_0, \ldots, w_{2n}) = \sum_{q=0}^{2n} \Phi(w_q)$ where each $w_q = \sum_{p=1}^{n} \psi_{q,p}(x_p)$ is itself a function of x. If the principal knew the value w_q for a random index q—then she could use the computational contract from Theorem 1 to incentivize the agent to reveal $f(x) = \widetilde{f}(w) = \sum_{q=0}^{2n} \Phi(w_q)$ using the mechanism $\mathcal{M}_1$.

However, the principal does not know the value of $w_q(x)$. Since $w_q(x)$ is itself a boundedly separable function of x, the principal might attempt the following mechanism

Round 1. Use mechanism $\mathcal{M}_1$ to incentivize the agent to reveal $\widetilde{f}(w(x)) = \sum_{q=0}^{2n} \Phi(w_q(x))$. This mechanism needs to query $w_q(x)$ for a uniformly random q. To obtain $w_q(x)$, go to round 2.

Round 2. Use mechanism $\mathcal{M}_1$ to incentivize the agent to reveal the boundedly separable function $w_q(x) = \sum_{p=1}^{n} \psi_{q,p}(x_p)$ by querying a uniformly random input coordinate p.

The problem with this approach is that the agent gets two rewards: one for announcing $\widetilde{f}(w)$ and one for announcing $w_q(x)$. Accordingly, it is possible that the agent would lie about w_q (thus, getting a lower reward in round 2), to manipulate the mechanism in round 1 and receive a higher reward overall.

The way to avoid this problem is to make the reward from round 2 so high that the agent has no incentive in round 2 to reveal a value v_q whose distance from $w_q(x)$ is greater than δ, for some δ that we will choose. We will argue that the agent will be incentivized in step 1 to announce $\widetilde{f}(v_1, \dots, v_n) = \sum_{q=0}^{2n} \Phi(v_q)$, instead of the true value $\widetilde{f}(w_1, \dots, w_n) = \sum_{q=0}^{2n} \Phi(w_q)$. Nevertheless, by using the uniform continuity of $\widetilde{f}$, we will guarantee that the agent's announcement is ϵ-close to the true value $\widetilde{f}(w) = f(x)$.

The Contract $\mathcal{M}_2$. We formalize the above intuition via the following contract and analysis.

Mechanism $\mathcal{M}_2 = (\mathcal{D}, \mathcal{S}, R)$

A play of $\mathcal{M}_2$ proceeds as follows

- In stage 1, the agent announces z, allegedly in the set $[f(x) - \epsilon, f(x) + \epsilon]$
- In stage 2, the mechanism announces q, drawn from $\mathcal{D}_1 = Uniform(\{0, \dots, 2n\})$
- In stage 3, the agent announces v_q, allegedly close to $\sum_{p=1}^{n} \psi_{p,q}(x_p)$
- In stage 4, the mechanism announces p, drawn from $\mathcal{D}_2 = Uniform(\{1, \dots, n\})$.

Given the message vector $m = (z, q, v_q, p)$, define

- $\mathcal{S}(m) = \{p\} \subset [n]$, and
- $R(m, x_{\mathcal{S}(m)})$ the value computed as follows
  - Let $\omega_1(v_q)$ be a random variable which is
    equal to 1 with probability $\frac{\Phi(v_q)+C}{2C}$ and
    equal to 0 with probability $1 - \frac{\Phi(v_q)+C}{2C}$.
  - Let ω_2 be a random variable which is
    equal to 1 with probability $\frac{\psi_{p,q}(x_p)}{2B}$ and
    equal to 0 with probability $1 - \frac{\psi_{p,q}(x_p)}{2B}$.
  - Interpret z as a random variable Ω_z which is
    equal to 1 with probability $\frac{z+(2n+1)C}{(2n+1)2C}$ and
    equal to 0 with probability $1 - \frac{z+(2n+1)C}{(2n+1)2C}$
  - Interpret v_q as a random variable Ω_{v_q} which is
    equal to 1 with probability $\frac{v_q+nB}{2Bn}$ and
    equal to 0 with probability $1 - \frac{v_q+nB}{2Bn}$.
  - Return the reward

$$R(m, x_{\mathcal{S}(m)}) = R((z, q, v_q, p), x_p) = BSR(\Omega_z, \omega_1) + \theta \cdot BSR(\Omega_{v_q}, \omega_2)$$

where $\theta > 0$ is a constant, which we determine later.

ϵ-INCENTIVE COMPATIBILITY. The reward function depends on z only through $BSR(\Omega_z, \omega_1(v_q))$. Thus—taking his choice of v_q in stage 3 as given—the agent is incentivized to reveal z in stage 1 such that the distribution Ω_z equals the distribution from which ω_1 is drawn. We have that

$$Pr(\omega_1 = 1) = \sum_{q=0}^{2n} Pr(q)Pr(\omega_1 = 1|q) = \frac{1}{2n+1}\sum_{q=0}^{2n} \frac{\Phi(v_q) + C}{2C}$$

$$= \frac{1}{(2n+1)2C}(\sum_{q=0}^{2n} \Phi(v_q) + (2n+1)C).$$

Since $Pr(\Omega_z = 1) = \frac{z+(2n+1)C}{(2n+1)2C}$, we conclude that $Pr(\Omega_z = 1) = Pr(\omega_1 = 1)$ if and only if

$$z = \sum_{q=0}^{2n} \Phi(v_q).$$

Note that z is not necessarily the true value $f(x_1, \dots, x_n) = \sum_{q=0}^{2n} \Phi(\sum_{p=1}^{n} \psi_{p,q}(x_p))$, since $v_q \neq \sum_{p=1}^{n} \psi_{p,q}(x_p)$. Note furthermore that the agent *is not incentivized* to announce $v_q = \sum_{p=1}^{n} \psi_{p,q}(x_p)$, since v_q enters his reward both via Ω_{v_q} and via ω_1, which is defined in terms of v_q. While announcing $v_q \neq \sum_{p=1}^{n} \psi_{p,q}(x_p)$ always decreases the term $\theta BSR(\Omega_{v_q}, \omega_2)$ in the agent's reward, it may increase the term $BSR(\Omega_z, \omega_1)$ enough to make such a deviation profitable.

We now give a bound on how much the agent can profit by announcing v_q instead of $w_q = \sum_{p=1}^{n} \psi_{p,q}(x_p)$. There are two effects that this deviation has on the expected reward:

- The first term of the expected reward changes by $\underset{\omega_1}{\mathbb{E}}[BSR(\Omega_z, \omega_1(v_q)) - BSR(\Omega_z, \omega_1(w_q)] \leq 3$ since the Brier scoring rule is bounded above by 3 and below by 0.
- Given a fixed q, by lemma 1, the second term of the expected reward changes by

  $$\theta \cdot \underset{\omega_2}{\mathbb{E}}[BSR(\Omega_{v_q}, \omega_2) - BSR(\Omega_{w_q}, \omega_2)] = -\frac{2\theta}{(2Bn)^2}(v_q - w_q)^2.$$

  Taking expectations over q, which is drawn uniformly from $\{0, 1, \dots, 2n\}$, the total expected loss in reward is $-\frac{2\theta}{(2Bn)^2}\frac{1}{2n+1}\|v - w\|_2^2$.

This analysis shows that when the agent reports v such that $\|v - w\|_2^2 > \delta$, the change in his reward is bounded by

$$3 - \frac{2\theta}{(2n+1)(2Bn)^2}\delta.$$

When θ is high enough, this change in reward is always negative and the agent always prefers not to announce a value v far away from w. In particular, if we set

$$\theta > \frac{3(2n+1)(2Bn)^2}{2\delta},$$

the agent is incentivized to report v such that $\|v - w\|_2^2 \leq \delta$.

Finally, we use the above analysis to show that the principal can incentivize the agent to announce z such that $|z - f(x)| < \epsilon$. For the desired approximation factor ϵ, let $\delta(\epsilon)$ be such that when $\|v - w\|_2^2 < \delta(\epsilon)$, we have $|\sum_{q=0}^{2n} \Phi(v_q) - \sum_{q=0}^{2n} \Phi(w_q)| < \epsilon$.[8] Since the agent is incentivized to announce v such that $\|v - w\|_2^2 < \delta(\epsilon)$ when $\theta > \frac{3(2Bn)^2}{2\delta(\epsilon)}$ and he is always incentivized to announce $z = \sum_{q=0}^{2n} \Phi(v_q)$, we conclude that the agent is always incentivized to announce z such that $|z - f(x_1, \dots, x_n)| < \epsilon$.

Case 2: $\mathbf{c}(\cdot) \not\equiv \mathbf{0}$ The above argument only applies when the agent's cost function c is identically 0. When this is not the case, we prove that we can scale the reward $R((z, q, v_q, p), x_p)$ by a large enough constant κ so that the agent is incentivized to learn enough inputs from the vector $(x_1, \dots, x_n)$ in order to provide an ϵ-approximation to f. As mentioned above, once the agent has learned this ϵ-approximation, the scoring rule guarantees that he will maximize his reward by reporting it to the mechanism. In contrast with the proof of Theorem 1, the agent

- does not need to learn the whole input $(x_1, \dots, x_n)$, since he only needs to give an approximation to $f(x)$, and
- learns his set of inputs S_A in two stages: by querying $S_{A,1}$ in round 1 of the mechanism and querying $S_{A,2}(z, q, x_{S_{A,1}})$ in round 2, after making his first-round announcement z, and learning the mechanism's random choice q and the partial input $x_{S_{A,1}}$. The final set of inputs S_A that the agent learns is the union $S_{A,1} \cup S_{A,2}$.

Our argument proceeds by analyzing the agent's optimization using backwards induction. For any partial inputs $x_{S_{A,1}}$, $x_{S_{A,2}}$ that the agent could learn, any z that the agent announces in stage 1, and any random q that the mechanism could draw in stage 2, let

$$v_q^*(z, x_{S_{A,1}}, x_{S_{A,2}}) \in \operatorname*{argmax}_v \mathop{\mathbb{E}}_{x \leftarrow X | (x_{S_{A,1}}, x_{S_{A,2}}), p \leftarrow \mathcal{D}_2} \left[R((z, q, v, p), x_p)\right]$$

$$\nu_2(z, q, x_{S_{A,1}}, S_{A,2}) = \mathop{\mathbb{E}}_{x \leftarrow X | x_{S_{A,1}}, p \leftarrow \mathcal{D}_2} \left[R((z, q, v_q^*(x_{S_{A,1}}, q, x_{S_{A,2}}), p), x_p)\right]$$

so that $\nu_2(z, q, x_{S_{A,1}}, S_{A,2})$ is the reward that the agent ultimately expects to receive when he learns $x_{S_{A,1}}$ and announces z in stage 1, receives q from the mechanism in stage 2, and chooses $S_{A,2}$ in stage 3. For any $z, x_{S_{A,1}}$ and any q, let

$$S_{A,2}^*(z, q, x_{S_{A,1}}) \in \operatorname*{argmax}_{S_{A,2}} \nu_2(z, q, x_{S_{A,1}}, S_{A,2}).^9$$

Proceeding by backwards induction, for any set $S_{A,1}$ we define

$$z^*(x_{S_{A,1}}) \in \operatorname*{argmax}_z \mathop{\mathbb{E}}_{x \leftarrow X | x_{S_{A,1}}, q \leftarrow \mathcal{D}_1} \left[\nu_2(z, q, x_{S_{A,1}}, S_{A,2}^*(z, q, x_{S_{A,1}}))\right],$$

[8]Note that $\delta(\epsilon)$ does not depend on x since $\sum_{q=0}^{2n} \Phi(\cdot)$ is uniformly continuous.

[9]There may be multiple such sets, but we shall argue that this multiplicity does not matter for our argument.

$$\nu_1(S_{A,1}) = \mathop{\mathbb{E}}_{x \leftarrow X, q \leftarrow \mathcal{D}_1}[\nu_2(z^*(x_{S_{A,1}}), q, x_{S_{A,1}}, S^*_{A,2}(z, q, x_{S_{A,1}})]$$

so that $\nu_1(S_{A,1})$ is the agent's expected reward when he chooses set $S_{A,1}$.

Let $S^*_{A_1} \in \operatorname{argmax}_S \nu_1(S)$. From case 1, we can infer that

- For any $S_{A,1}$, the agent's choice of set $S^*_{A,2}(z^*(x_{S_{A,1}}), q, S_{A,1})$ in stage 3 will give him enough information to report v_q such that $\|v - w\|_2^2 < \delta$.
- Given that the agent is reporting such a v_q in stage 3, the agent chooses $S^*_{A,1}$ in stage 1 to get enough information to learn $z = z^*(x_{S_{A,1}})$ such that $z = \sum_{q=0}^{2n} \Phi(v_q)$.

We now proceed to scale the reward so that, even when there is cost, the agent is always incentivized to choose $S^*_{A,1}$ in stage 1 and is incentivized to choose $S^*_{A,2}(\cdot, \cdot, \cdot)$ in stage 3. Choose $\kappa > 0$ such that

- For any $S_{A,1} \neq S^*_{A,1}$ and any $S_{A,2}$, we have
$$\kappa \cdot \nu_1(S^*_{A,1}) - c(|S^*_{A,1}| + |S_{A,2}|) > \kappa \cdot \nu(S_{A,1}) - c(|S_{A,1}| + |S_{A,2}|)$$
- For any z, any q and any $x_{S_{A,1}}$, and any $S_{A,2} \neq S^*_{A,2}(z, q, S_{A,1})$
$$\kappa\nu_2(z, q, x_{S_{A,1}}, S^*_{A,2}(z^*(x_{S_{A,1}}), q, S_{A,1})) - c(|S_{A,1}| + |S^*_{A,2}(z, q, S_{A,1}))|) >$$
$$\kappa\nu_2(z, q, x_{S_{A,1}}, S_{A,2}) - c(|S_{A,1}| + |S_{A,2}|)$$

Such a κ exists because $S^*_{A,2}(z, q, S_{A,1})$, $S^*_{A,1}$ are chosen to maximize ν_1 and ν_2, respectively. Let $\widetilde{R}(\cdot, \cdot) \overset{\text{def}}{=} \kappa \cdot R(\cdot, \cdot)$ be a scaled reward function. Let $\widetilde{\nu_1} \overset{\text{def}}{=} \kappa \cdot \nu_1, \widetilde{\nu_2} \overset{\text{def}}{=} \kappa \cdot \nu_2$ Then, by our choice of κ, we have

- For any $S_{A,1} \neq S^*_{A,1}$ and any $S_{A,2}$, we have
$$\widetilde{\nu_1}(S^*_{A,1}) - c(|S^*_{A,1}| + |S_{A,2}|) > \widetilde{\nu}(S_{A,1}) - c(|S_{A,1}| + |S_{A,2}|)$$
- For any z, any q and any $x_{S_{A,1}}$, and any $S_{A,2} \neq S^*_{A,2}(z^*(x_{S_{A,1}}), q, S_{A,1})$
$$\widetilde{\nu_2}(z, q, x_{S_{A,1}}, S^*_{A,2}(z^*(x_{S_{A,1}}), q, S_{A,1})) - c(|S_{A,1}| + |S^*_{A,2}(z^*(x_{S_{A,1}}), q, S_{A,1}))|) >$$
$$\widetilde{\nu_2}(z, q, x_{S_{A,1}}, S_{A,2}) - c(|S_{A,1}| + |S_{A,2}|).$$

Thus, by changing the reward of function of $\mathcal{M}_2$ to be $\kappa \cdot R$, we can incentivize the agent to gather enough information to be able to report v such that $\|v - w\|_2^2 < \delta$ and z such that $|z - f(x)| < \epsilon$.

We conclude by noting that

- We can further increase κ to ensure individual rationality.
- Even if there are multiple optimal choices for $S^*_{A,1}$ and $S^*_{A,2}$, all such choices guarantee that the agent gathers enough information to report z within ϵ of $f(x)$.

Q.E.D.

4.3 Our Fourth Theorem

Our mechanism $\mathcal{M}_2$ queries only one coordinate x_p from the input vector, and uses two rounds of interaction between the principal and the agent. This raises the question of whether there exist *one round* contracts which can be used to approximately delegate continuous functions and query few inputs. We show in the following theorem that this is not the case.

Theorem 4. *There exists a continuous function $f : [0,1]^n \to \mathbb{R}$ such that every one-round, ϵ-incentive compatible contract $\mathcal{M} = (\mathcal{D}, R)$ must query the whole input $(x_1, \ldots, x_n)$.*

The proof of Theorem 4 is given in appendix C.

5 Discussion

Optimality and Unlimited Liability In the above discussions we have been careful to minimize the number of queries that the mechanism makes to the input (as well as the number of interaction rounds between the principal and the agent). In addition to the query cost, the principal of course bears the monetary cost of paying the agent's reward. As we mentioned in the introduction, this monetary cost can be minimized for any of our mechanisms when the agent has unlimited liability. The principal can simply charge the agent a fixed fee equal to the agent's expected utility from participating in the mechanism. Since this fixed fee can be computed using expectations over X (rather than querying the actual input x), the principal can compute this reward at no extra cost. Since the principal minimizes her expected payment to the agent and the number of queries she makes, our contracts are optimal in the unlimited liability scenario.

In the classical setting, this type of argument is known as "selling the firm" [8] and makes the problem trivial. In this setting, the problem is still non-trivial because the principal still needs to be convinced that the agent has computed a correct approximation to $f(x)$ and therefore needs to query some set S of inputs in order to verify the agent's answer. Thus, even when we sell the firm and minimize the principal's monetary cost, the problem of minimizing the principal's query cost still stands.

Accounting for Algorithmic Cost In our results we assumed that data collection is expensive but, once the data is available, running an algorithm on the collected data is free. An advantage of this model over other ways of measuring computational complexity is that we can precisely analyze how many pieces of data a principal has to observe in order to incentivize the agent to evaluate a continuous function of the entire dataset. Indeed, we showed that the principal only needs to observe *a single piece* of data.

An alternative approach would be to assume that the data is already available (and thus cost free), while running algorithms on the available data has a cost that increases with the number of operations performed. We are also interested in delegating computation to a rational agent in this model. We believe that doing so is likely to require techniques from computer science, such as computationally sound interactive proof systems [14] [9]—where the agent gives a verifiable evidence to the principal that his answer is correct. This verifiable evidence guarantees to the principal that no malicious agent (who may go out of his way to cheat her) will be able to deceive her. Such guarantees are very strong but require very complicated protocols. We believe that by properly modeling the rationality of the agent, as we have done in this paper, we can vastly simplify these protocols.

References

[1] N. Al-Najjar, L. Anderlini and L. Felli Undescribable Events. *The Review of Economic Studies*, 73(4): 849-868, 2006.

[2] L. Anderlini and L. Felli Incomplete written contracts: Undescribable states of nature. *The Quarterly Journal of Economics*, 109(4):1085-1124, 1994.

[3] P. Azar and S. Micali Rational Proofs. *Symposium on the Theory of Computing.* 44:1017-1028, 2012.

[4] G.W. Brier. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 1950.

[5] R.T. Clemen Incentive contrats and strictly proper scoring rules. *Test* 11(1): 167-189,2002.

[6] T. Gneiting and A.E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

[7] I. J. Good. Rational Decisions. *Journal of the Royal Statistical Society*, Ser. B 14:107-114.

[8] B. Holmstrom . Moral hazard and observability. *The Bell Journal of Economics*, 10(1):74-91,1979.

[9] J. Kilian. A note on efficient zero-knowledge proofs and arguments. *ACM symposium on Theory of computing* 24: 723–732. ACM, 1992.

[10] A. N. Kolmogorov. On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition. *Amer. Math. Soc. Transl* 28: 55-59,1963.

[11] N.S. Lambert. Elicitation and evaluation of statistical forecasts. *Manuscript,* 2011.

[12] CERN Computing `http://home.web.cern.ch/about/computing`

[13] W.B. MacLeod. Complexity and contract. *Revue d'conomie Industrielle* 92(1): 149-178, 2000.

[14] S. Micali. Computationally sound proofs. *SIAM Journal on Computing*, 30(4):1253–1298, 2001.

[15] K. Osband. Optimal forecasting incentives. *The Journal of Political Economy* 97(5):1091-1112, 1989.

[16] S. Shavell. On moral hazard and insurance. *The Quarterly Journal of Economics,* 93(4): 541-562,1979.

Appendix

A Proof of Theorem 2

Theorem 2. *There exists a continuous, bounded and non-separable function $f : [0,1]^n \to \mathbb{R}$ such that any direct computational contract $\mathcal{M} = (\mathcal{D}, R)$ that is incentive compatible for f must query the whole input vector $(x_1, \ldots, x_n)$.*

Proof. First, we make some notation explicit. So far, we have denoted the reward of a direct mechansim as a function $R(z, x_S)$ of the agent's message z and the principal's queried partial input x_S. Implicit in this notation is that the reward also depends on the principal's choice of S. In this proof, we will find it helpful to make this dependence explicit by writing $R(z, x_S, S)$.

Let $f(x) = \sum_{i=1}^{n} \prod_{j \neq i} x_j = x_1 x_2 \ldots, x_{n-1} + x_2 x_3 \ldots x_n + x_n x_1 \ldots .x_{n-2}$. We show that any direct contract which is incentive compatible for f must query the whole vector $(x_1, \ldots, x_n)$. To show this, we proceed by contradiction. Let $k < n$ and assume there is a k-query incentive compatible contract $\mathcal{M} = (\mathcal{D}, R)$ for delegating f. We will show that exists a *symmetric* contract $\mathcal{M}_{sym} = (\mathcal{D}_{sym}, R_{sym})$ for delegating f which has the property that

$$R_{sym}(z, x_S, S) = R_{sym}(z, x_{S'}, S') \text{ for any } S, S' \text{ such that } x_S = x'_S \tag{3}$$

To see this, let $\sigma : [n] \to [n]$ be a permutation. Let $\sigma(S) = \{\sigma(i) : i \in S\}$ and define $f_\sigma(x_1, \ldots, x_n) = f(x_{\sigma(1)}, \ldots, x_{\sigma(n)})$. Since f is a symmetric function we have $f_\sigma \equiv f$.Since $\mathcal{M}$ is a mechanism that incentivizes the agent to reveal $f(x_1, \ldots, x_n)$, the following is a contract that incentivizes f_σ.

$\mathcal{M}_\sigma = (\mathcal{D}_\sigma, R_\sigma)$

- $\mathcal{D}_\sigma$ is such that $Pr(\mathcal{D}_\sigma = S) = Pr(\mathcal{D} = \sigma(S))$.
- For any z, x_S, the reward function $R_\sigma(z, x_S, S) = R(z, x_S, \sigma(S))$

We can use the mechanism $\mathcal{M}_\sigma$ to construct a symmetric mechanism, where the principal gives the same reward regardless of which set S she queries. The symmetric mechanism $\mathcal{M}_{sym}$

is constructed by taking a permutation σ at random and running the mechanism $\mathcal{M}_\sigma$. The mechanism $\mathcal{M}_{sym} = (\mathcal{D}_{sym}, R_{sym})$ satisfies the property (3) . Furthermore, $\mathcal{M}_{sym}$ queries all sets with equal probability.

Finally, to see that this leads to a contradiction, note that $\mathcal{M}_{sym}$ incentivizes the agent to reveal f (this is because each $\mathcal{M}_\sigma$ incentivizes the agent to reveal $f_\sigma \equiv f$). Thus, there exists at least one set S for which the reward $R_{sym}(f(x), x_S, S) > R(z, x_S, S)$ for $z \neq f(x)$. However, since $\mathcal{M}_{sym}$ is symmetric, we must have $R_{sym}(f(x), x_{S'}, S') > R_{sym}(z, x_{S'}, S')$ for all S' such that $x_S = x_{S'}$. We can construct the contradiction by considering the inputs $x = (x_1, x_2, x_3, \ldots, x_n) = (1, 1, 1, \ldots, 1)$ and $\widetilde{x} = (\widetilde{x_1}, \widetilde{x_2}, \widetilde{x_3}, \ldots, \widetilde{x_n}) = (1, 1, 1, \ldots, 1, 0)$. We have $f(x) = n$ and $f(\widetilde{x}) = 1$. Let $S = \{1, \ldots, n-1\}$ and note that $x_S = \widetilde{x}_S$. By the above argument, any set S must satisfy the two inequalities

$$R_{sym}(f(x), x_S, S) > R_{sym}(f(\widetilde{x}), x_S, S)$$

$$R_{sym}(f(\widetilde{x}), \widetilde{x}_S, S) > R_{sym}(f(x), \widetilde{x}_S, S).$$

Since $\widetilde{x}_S = x_S = (1, \ldots, 1)$, this leads to the contradiction

$$R_{sym}(f(x), \widetilde{x}_S, S) = R_{sym}(f(x), x_S, S) > R_{sym}(f(\widetilde{x}), S, x_S, S) =$$

$$R_{sym}(f(\widetilde{x}), \widetilde{x}_S, S) > R_{sym}(f(x), \widetilde{x}_S, S)$$

which concludes the proof. *Q.E.D.*

B Proof of Lemma 1

Lemma 1. *Let v, w be real numbers in $[0, 1]$ and let V, W be random variables over $\{0, 1\}$ such that $Pr(V = 1) = v, Pr(W = 1) = w$. Then*

$$\mathop{\mathbb{E}}_{\omega \leftarrow V}[BSR(V, \omega) - BSR(W, \omega)] = 2(v - w)^2$$

Proof. Note that we have $\|V\|_2^2 = v^2 + (1-v)^2$ and $\|W\|_2^2 = w^2 + (1-w)^2$ We have

$$\mathop{\mathbb{E}}_{\omega \leftarrow V} BSR(V, \omega) = Pr(\omega = 1)(2Pr(V = 1) - \|V\|_2^2 + 1) + Pr(\omega = 0)(2Pr(V = 0) - \|V\|_2^2 + 1)$$

$$= 2v^2 + 2(1-v)^2 - v^2 - (1-v)^2 + 1 = v^2 + (1-v)^2 + 1 = 2v^2 - 2v + 2.$$

We also have

$$\mathop{\mathbb{E}}_{\omega \leftarrow V} BSR(W, \omega) = Pr(\omega = 1)(2Pr(W = 1) - \|W\|_2^2 + 1) + Pr(\omega = 0)(2Pr(W = 0) - \|W\|_2^2 + 1)$$

$$= 2vw + 2(1-v)(1-w) - w^2 - (1-w)^2 + 1 = 4vw + 2 - 2v - 2w^2.$$

Taking differences, we have

$$\mathop{\mathbb{E}}_{\omega \leftarrow V}[BSR(V, \omega) - BSR(W, \omega)] = 2(v^2 + w^2) - 2v - 4vw + 2v = 2v^2 + 2w^2 - 4wv = 2(v - w)^2$$

Q.E.D.

C Proof of Theorem 4

Theorem 4. *There exists a continuous function $f : [0,1]^n \to \mathbb{R}$ such that every one-round contract $\mathcal{M} = (\mathcal{D}, R)$, that is ϵ-incentive compatible for f, must query the whole input $(x_1, \dots, x_n)$.*

Proof. Let $f(x) = \sum_{i=1}^n \prod_{j \neq i} x_j = x_1 x_2 \dots x_{n-1} + x_2 x_3 \dots x_n + x_n x_1 \dots .x_{n-2}$ as in Theorem 2. In Theorem 2, we showed that there is no direct contract for f that makes less than n queries to the input vector. The main difference between the setting in this proof and the proof in theorem 2 is that the agent announces a real number $z = m(x)$ instead of $f(x)$, and the principal recovers an approximation to $f(x)$ by applying a continuous function h such that $|f(x) - h(m(x))| < \epsilon$.

The proof now proceeds analogously to the proof of theorem 2. Let

$$m^*(x) = \underset{z}{\operatorname{argmax}} \underset{S \leftarrow \mathcal{D}}{\mathbb{E}}[R(z, x_S)].$$

For any $z \neq m^*(x)$, there must exist a set S

The proof now proceeds analogously to the proof of theorem 2, given above. Let σ be a permutation of $[n]$ and let $f_\sigma(x) = f(x_{\sigma(1)}, \dots, x_{\sigma(n)})$ and let $\mathcal{M}_\sigma$ be as in the proof of theorem 2. This mechanism now incentivizes the agent to reveal a message $m_\sigma(x)$ such that $|f_\sigma(x) - m_\sigma(x)| < \epsilon$. Since f is a symmetric function, we have $f_\sigma \equiv f$ and the mechanism incentivizes the agent to reveal $m_\sigma(x)$ such that $|f(x) - m_\sigma(x)| < \epsilon$.

Let $\mathcal{M}_{sym}$ be the mechanism which chooses a permutation σ uniformly at random and runs $\mathcal{M}_\sigma$. Let $m_{sym}(x) = \operatorname{argmax}_z \mathbb{E}_{S \leftarrow \mathcal{D}_{sym}}[R_{sym}(z, x_S)]$ and note that $m_{sym}(x) \in [min_\sigma m_\sigma(x), max_\sigma m_\sigma(x)]$.[10] Since $m_\sigma(x)$ is within ϵ of $f(x)$ for every σ, we have $|m_s ym(x) - f(x)| < \epsilon$, so the agent is incentivized to give an ϵ-approximation to f.

We now proceed as in the proof of Theorem 2, writing the reward R_{sym} explictly as $R_{sym}(z, x_S, S)$ and recalling that

$$R_{sym}(z, x_S, S) = R_{sym}(z, x_{S'}, S') \text{ for any } S, S' \text{ such that } x_S = x'_S$$

We can construct a contradiction by considering the inputs $x = (x_1, x_2, x_3, \dots, x_n) = (1, 1, 1, \dots, 1)$ and $\widetilde{x} = (\widetilde{x_1}, \widetilde{x_2}, \widetilde{x_3}, \dots, \widetilde{x_n}) = (1, 1, 1, \dots, 1, 0)$. We have $f(x) = n$ and $f(\widetilde{x}) = 1$. Let $S = \{1, \dots, n-1\}$ and note that $x_S = \widetilde{x}_S$. Let $z \in [f(x) - \epsilon, f(x) + \epsilon]$ and $\widetilde{z} \in [f(\widetilde{x}) - \epsilon, f(\widetilde{x}) + \epsilon]$, and note that (unless ϵ is very large), z is always different than $\widetilde{z}$. By the above argument, any set S must satisfy the two inequalities

$$R_{sym}(z, x_S, S) > R_{sym}(\widetilde{z}, x_S, S)$$

[10]For illustration purposes, we assume in this footnote that R is differentiable with respect to z. Then each $R_\sigma(z, x) = \mathbb{E}_{S \leftarrow \mathcal{D}_\sigma}[R_\sigma(z, x_S)]$ is a concave function of z. Thus, $\frac{\partial R_\sigma}{\partial z}(m_\sigma(x), x) = 0$ and decreases with z. Let $m_{min}(x) = min_\sigma m_\sigma(x)$ and $m_{max}(x) = max_\sigma m_\sigma(x)$. For every σ, we have $\frac{\partial R_\sigma}{\partial z}(m_{min}, x) \leq 0$ and $\frac{\partial R_\sigma}{\partial z}(m_{max}, x) \geq 0$. The function $R_{sym}(z, \cdot) = \mathbb{E}_\sigma R_\sigma(z, \cdot)$ is a convex combination of the concave functions R_σ, and is therefore concave. Since derivatives are additive, we have $\frac{\partial R_{sym}}{\partial z}(m_{min}, x) \leq 0$ and $\frac{\partial R_{sym}}{\partial z}(m_{max}, x) \geq 0$. Since R_{sym} is concave in z, this means that $m_{sym}(x) \in [m_m in(x), m_m ax(x)]$. When R_σ is not differentiable (but still concave), we can apply the same argument using subgradients instead of derivatives.

$$R_{sym}(\widetilde{z}, \widetilde{x}_S, S) > R_{sym}(z, \widetilde{x}_S, S).$$

Since $\widetilde{x}_S = x_S = (1, \ldots, 1)$, this leads to the contradiction

$$R_{sym}(z, \widetilde{x}_S, S) = R_{sym}(z, x_S, S) > R_{sym}(\widetilde{z}, S, x_S, S) =$$

$$R_{sym}(\widetilde{z}, \widetilde{x}_S, S) > R_{sym}(z, \widetilde{x}_S, S).$$

From this contradiction, we conclude that no one-round mechanism that makes $k < n$ queries can be ϵ-incentive compatible for f.

Q.E.D.