# Saving Money While Polling with InterPoll using Power Analysis

**Benjamin Livshits and Todd Mytkowicz**
Microsoft Research

### Abstract

Crowd-sourcing is increasingly being used for large-scale polling and surveys. Companies such as SurveyMonkey and Instant.ly make crowd-sourced surveys commonplace by making the crowd accessible through an easy-to-use UI with easy to retrieve results. Further, they do so with a relatively low latency by having dedicated crowds at their disposal.

In this paper we argue that the ease with which polls can be created conceals an inherent difficulty: the survey maker does not know how many workers to hire for their survey. Asking too few may lead to samples sizes that "do not look impressive enough." Asking too many clearly involves spending extra money, which can quickly become costly. Existing crowd-sourcing platforms do not provide help with this, neither, one can argue, do they have any incentive to do so.

In this paper, we present a systematic approach to determining how many samples (i.e. workers) are required to achieve a certain level of statistical significance by showing how to automatically perform power analysis on questions of interest. Using a range of queries we demonstrate that power analysis can save significant amounts of money and time by often concluding that only a handful of results are required to arrive at a decision.

We have implemented our approach within InterPoll, a programmable developer-driven polling system that uses a generic crowd (Mechanical Turk) as a back-end. InterPoll automatically performs power analysis by analyzing both the structure of the *query* and the *data* that it dynamically polls from the crowd. In all of our studies we obtain statistically significant results for under $30, with most costing less than $10. Our approach saves both time and money for the survey maker.

## Introduction

Online surveys are a powerful force for assessing properties of the general population and are popular for marketing studies, product development studies, political polls, customer satisfaction surveys, and medical questionnaires. Online polls are widely recognized as an affordable alternative to in-person surveys, telephone polls, or face-to-face interviews. Psychologists have argued that online surveys are far superior to the traditional approach of finding subjects which involves recruiting college students, leading to the famous quip about psychology being the "study of the college sophomore" (Cooper, McCord, and Socha 2011).



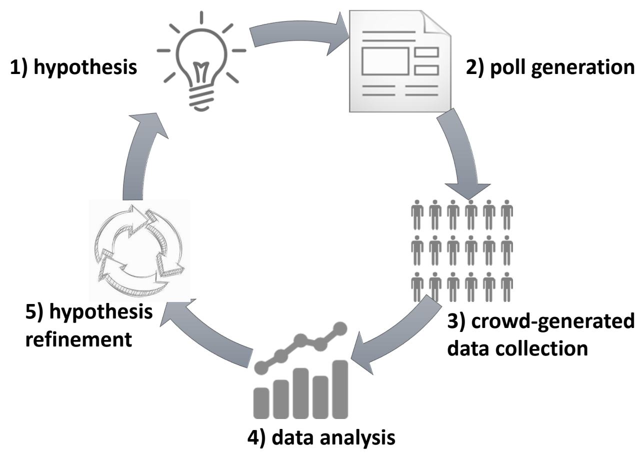



Figure 1: InterPoll: system architecture.

**Crowd-sourced polling:** The focus of this paper is on crowd-sourced polling with InterPoll and a brief summary of how InterPoll works is shown in Figure 1. The polling process begins by

1. *formulating* a hypothesis and then using queries to
2. *generate* polls that are sent to the crowd to
3. produce *data* that can be
4. *analyzed* to (optionally) produced
5. further, *refined* hypotheses.

In this paper we report on our experiences of both coding up polls using language-integrated queries (LINQ) in C# and analyzing those polls on a Mechanical Turk-based back end. We should point out that unlike much crowd-sourced work, the focus of InterPoll is on inherently *subjective* opinion polls that lack an clear notion of objective truth. As such, we are not focusing as much on the issue of dealing with cheaters, etc.

**How many is good enough?** Online surveys allow one to reach wider audience groups and to get people to answer questions that they may not be comfortable responding in a face-to-face setting. While online survey tools such as Instant.ly, SurveyMonkey, Qualtrics, and Google Customer Surveys take care of the mechanics of online polling and make it easy to get *started*, the results they produce often create more questions than the answers they provide (Couper 2000; Evans, Hempstead, and Mathur 2005; Dillman, Tortora, and Bowker 1998; Gunn 2002; Wyatt 2000; Keeter, Christian, and Researcher 2012).

**Cost:** Of course, the number of workers directly translates to the ultimate *cost* of the poll, which is an important if not the most important consideration for poll makers, especially given that thousands of participants may be required. Even if answering a single question can costs cents, often getting a high level of assurance for targeted population segment involves hundreds of survey takers and thus significant high cost; for example, in the case of Instant.ly, $5–7 per survey completion is not uncommon. One of the key long-term goals of InterPoll is to reduce the end-to-end cost of polling and thereby democratize access to human-generated opinion data.

**Beyond the status quo:** While putting together an online poll using a tool such as SurveyMonkey is not very difficult, a fundamental question is how many people to poll. Indeed, polling too few yields results that are not statistically significant; polling too many is a waste of money. None of the current survey platforms help the survey-maker with deciding on the appropriate number of samples. Today's online survey situation can perhaps be likened to playing slot machines with today's survey sites playing the role of a casino; it is clearly in the interest of these survey sites to encourage more polls being completed. Our aim is to change this situation and reduce the number of samples needed for statistically significant approach to decision making.

**Contributions:** To summarize, this paper makes the following contributions.

- We show how to formulate a wide range of decision problems about people as crowd-sourced queries expressed in LINQ, making it possible to embed them in larger applications.
- Given a query, we automatically perform power analysis, limiting the number of answers we need to get from the crowd.
- Using a range of queries we show that power analysis is effective in keeping the number of required samples down, especially compared to alternatives such as Chernoff bounds.

## The Design of InterPoll

At the core of InterPoll is a marriage of two technologies: language integrated queries, or LINQ, which lets programmers easily express SQL like queries over a data-source, in this case a crowd, and `Uncertain⟨T⟩`, a programming language abstraction which lets developers accurately compute with distributions (Bornholt, Mytkowicz, and Mckinley ). By exploiting these two abstractions, programmers can easily express polls in code and InterPoll's runtime manages the complexity that comes along with both issuing that poll *and* dealing with any resulting computation on its result. InterPoll treats each response to a LINQ poll as a *sample* from a population. As such, each poll is accurately characterized as a distribution over a set of responses. `Uncertain⟨T⟩` is a generic type which lifts operations over $T$ to distributions over $T$ and thus all InterPoll polls are easily converted to `Uncertain⟨T⟩` responses.

In this section we present a series of motivating examples to familiarize the reader with InterPoll. In particular, we first demonstrate how queries can be encoded as LINQ statements and then show how the response to such queries can be easily represented as distributions by exploiting `Uncertain⟨T⟩`.

### LINQ to Crowd: Declarative Queries

One goal of InterPoll is to democratize crowd-sourced polls by letting developers express polls in code. We accomplish this by using LINQ (Mayo 2008), language-integrated queries. LINQ is natively supported by .NET, with Java providing similar facilities with JQL. Relying on LINQ allows for easy integration between computer and human computation and obviates the need for domain-specific languages. One of the main benefits of LINQ is that it is *lazy* by construction and as such queries do not execute until programmers want to act on that data (e.g., printing out the results of the query, or comparing the results of two queries).

**Example 1 (Basic data collection)** A simple poll in InterPoll:

```
var people = GetPeople("Height survey",...);
var height = (from person in people
    select new
    {
        Height = person.PoseQuestion<int>(
          "What is your height, in centimeters?"),
        Gender = person.Gender,
        Ethnicity = person.Ethnicity,
    })
```

The first line gets a handle to a population of users, in this case obtained from Mechanical Turk, although other back-ends are also possible. Populations on which we operate have associated demographic information; for example, InterPoll represents every response to this poll by a C# anonymous type with three fields: `Height`, which is an `int` and represents the response to the textual question as arguments to `person.PoseQuestion`, and `Gender` and `Ethnicity`, both of which are built in enumerations and represent demographic information of the population. The result of this LINQ query is an `IEnumerable`, with each item being a single response to this query. □

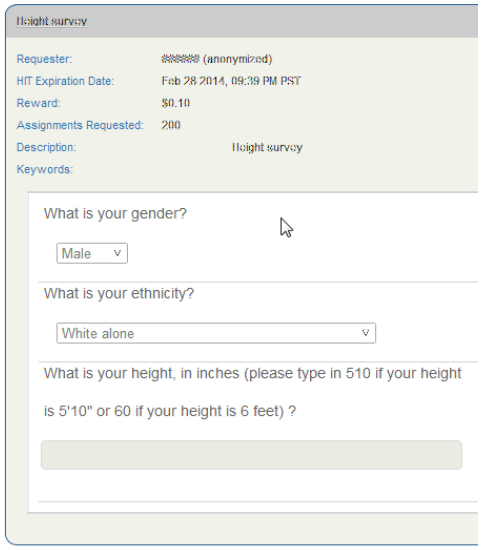



Figure 2: An automatically generated HIT from Example 1.

InterPoll automatically compiles a query to a XML form which is then communicated to the Mechanical Turk backend as a new HIT. An example of a such a form for the above query is shown in Figure 2. Our backend monitors this job and polls for more by expanding the number of outstanding assignments as required.

**Example 2 (Filtering)** Given the prior poll, it is possible to do a subsequent operations on the results, such as filtering.

```
var females = from person in height
  where person.Gender == Gender.FEMALE
  select person.Height;

var males = from person in height
  where person.Gender == Gender.MALE
  select person.Height;
```

The code above filters the previously collected population into two groups, one consisting of men and the other of women. It is important to realize that both of these queries represent enumerations of heights, in centimeters. For instance, the first handful of values for females yields the following list of heights: 173, 162, 165, 162, 162, 157, 167, 165, 157, 165, 160, 158, 150, while for males, the list looks like this 186, 180, 182, 187, 175, 180, 180, 190, 183, 177, 173, 188, 175, 170, 180, 170, 178, 190, 183, 172, 170, 187, 175, 1 0, 191, 198, 175, 175, 180, 176, 164, 193, 160, 175, 175, 175, 176. Eyeballing the two lists and computing the means, one can get the intuition that in general males are taller than females. □

**Example 3 (From LINQ to Uncertain⟨T⟩)** After expressing a query via LINQ, programmers often want to compute on the result. For example, the suppose a programmer wants to know if men are more likely than not to be $10cm$ taller than women.

```
1 Uncertain<int> maleHeight = males.ToRandomVariable();
2 Uncertain<int> femaleHeight = females.ToRandomVariable();
3 if (maleHeight + 10 > femaleHeight)
4   Console.WriteLine("Males are 10cm taller than females");
```

Line 1 and 2 transform the query from an `IEnumerable` over integers into a *distribution* over integers, where each element of the enumeration is an independent and identically distributed sample from that distribution. Likewise, line 3 adds the constant (or point-mass distribution) to the heights of males and then compares that, using the less than operator, to the height of females.

The next section discusses explicitly how the InterPoll runtime uses `Uncertain⟨T⟩` to (1) compute with distributions (i.e., adding $10cm$ to every male height) and (2) use acceptance sampling to bound the number of samples required when comparing the two distributions `maleHeight` + 10 and `femaleHeight`. □

## `Uncertain⟨T⟩`: Computing with Distributions

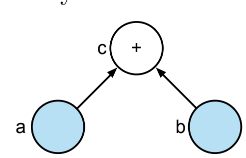
`Uncertain⟨T⟩` frees novice programmers from the burden of computing with data that is accurately expressed as distributions. Because it is a generic type with the appropriate operator overloading, novice developers compute with an `Uncertain⟨T⟩` as they would with a normal $T$. Computing with distributions of $T$, rather than single elements of $T$, does add overhead but the `Uncertain⟨T⟩` runtime is specifically designed to mitigate this overhead through a set of novel optimizations and by exploiting sampling.

Under the hood, the `Uncertain⟨T⟩` runtime abstracts a program into a Bayesian network representation of a program and then samples from that network at conditionals in order to evaluate *evidence* for the condition. A Bayesian network is a directed, acyclic graphical model — nodes in the graph represent random variables from a program and edges between nodes represent conditional dependencies between those random variables.

This representation lifts the concrete semantics of the original program into a probabilistic one in which all program variables are distributions. Constants (e.g., x = 3) are point-mass distributions and "known distributions" (e.g., uniform, Gaussian, programmer specified) are symbolic representations. The program induces further distributions by computing with the former two. For example, the following code

```
    Uncertain<double> a = new Gaussian(4, 1);
    Uncertain<double> b = new Gaussian(5, 1);
    Uncertain<double> c = a + b;
```

results in a simple Bayesian network show below



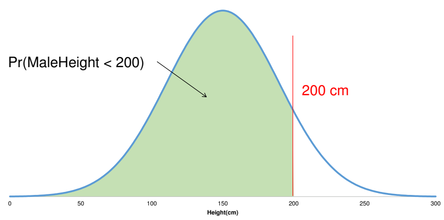



Figure 3: A probabilistic decision about the height of males on Amazon's Mechanical Turk.

with three nodes that represents the computation $\texttt{c} = \texttt{a} + \texttt{b}$. `Uncertain⟨T⟩` evaluates this Bayesian network when it needs the distribution of `c` (e.g., at a conditional when the programmer wants to branch on the value of `c`), which depends on the distributions of `a` and `b`.

### Decisions with Distributions

Programs eventually act on their data, usually in the form of conditionals. How do we accurately evaluate a conditional when a program variable is a distribution?

`Uncertain⟨T⟩` defines the semantics of a conditional expression over an probabilistic conditional variable by computing *evidence* for a conclusion. For example, suppose we want to know if males are less than $200cm$ tall?

```
Uncertain<double> maleHeight = ...
if (maleHeight < 200) Console.Write("Male height < 200cm");
```

A non-`Uncertain⟨T⟩` conditional asks: "is `maleHeight` less than $200cm$?" which is difficult, given `maleHeight` is an estimate. In contrast, the `Uncertain⟨T⟩` runtime implicitly converts the above conditional into the question: "how much evidence is there that `maleHeight` is less than $200cm$?".

The *evidence* that males are less than $200cm$ tall is $\Pr[\texttt{maleHeight} < 200]$, the shaded area in Figure 3. Evaluating a conditional implies accurately sampling and estimating this area of the plot.

### Combining the Pieces

Example 3 provides a full example of InterPoll. A LINQ query asks males and females, respectively for their heights, a single call which turns those LINQ queries into `Uncertain⟨int⟩` responses, a computation over the male heights which adds the point-mass distribution of $10cm$ to the male heights, and then, finally, a comparison which asks whether men are more likely than not to be $10cm$ taller than women. A key benefit of such a system is that it has all the information necessary in hand when it judges each decision point, or conditional and only needs to take as many samples such that it can accurately evaluate that judgment.

The next section discusses how `Uncertain⟨T⟩` accurately and efficiently implements decisions with hypothesis tests, only drawing enough samples so as to answer a particular question and no more, thus saving costs for the poll writer.

## Power and Confidence

In order to evaluate evidence at conditionals, the `Uncertain⟨T⟩` runtime uses acceptance sampling, a form of hypothesis testing. For example, in Example 3, the `Uncertain⟨T⟩` runtime automatically sets up a t-test with the null-hypothesis

$$H_0 : \Pr[\texttt{maleHeight} + 10 \geq \texttt{femaleHeight}]$$

and the alternate

$$H_A : \Pr[\texttt{maleHeight} + 10 < \texttt{femaleHeight}]$$

where $\Pr[\texttt{maleHeight} + 10]$ and $\Pr[\texttt{femaleHeight}]$ is the random variable representation of the program variables $\texttt{maleHeight} + \texttt{10}$ and `femaleHeight`. Each conditional in an `Uncertain⟨T⟩` program is designed to evaluate this hypothesis test by sampling from the condition variable in order to either reject $H_0$ (and thus accept $H_A$) or vice-versa.

### Bayesian Network

Recall every `Uncertain⟨T⟩` variable is a Bayesian network and as such each variable is dependent *only* on its parents. Thus, sampling the root node consists of generating a sample at each leaf node and propagating those values through the graph.

`Uncertain⟨T⟩`'s sampling and hypothesis testing is designed to mitigate against two potential sources of error:

1. the probability that we obtain a good estimate of the evidence (i.e., our *confidence* in the estimate) and
2. the extent to which our estimate of the evidence is *accurate*.

All conditional variables are logical properties over random variables and therefore *Bernoulli* random variables. `Uncertain⟨T⟩` exploits C#'s implicit conversions to implicitly cast from a *Bernoulli* to a `bool`. The mechanism behind this cast is a hypothesis test. `Uncertain⟨T⟩` explicitly lets a programmer control the likelihood of a good estimate—or her *confidence*—by increasing $\alpha$ when calling into the hypothesis test. Likewise, a programmer can control the *accuracy* of her estimate by decreasing $\epsilon$. In concert, these parameters let a programmer trade off false-positives and false-negatives with sample size.

### Sequential Acceptance

`Uncertain⟨T⟩` performs a hypothesis test using Wald's sequential probability ratio test (SPRT)(Wald 1945) to dynamically choose the right sample size for any conditional, only taking as many samples as necessary to obtain a statistically significant result *with* an appropriate power. SPRT is designed to balance false-positives (i.e., based on a *confidence* level) and false-negatives (i.e., based on the *accuracy*, or power of the test).

Assume $X_i \sim Bernoulli(p)$ is an independent sample of a condition variable where $p$ is the *true* probability of the condition variable (and unknown). For example, in Figure 3, the Uncertain⟨T⟩ runtime draws "true" samples proportional to the shaded area (and "false" samples proportional to the unshaded area). Let $X = X_1 + X_2 + \cdots + X_n$ be the sum of $n$ independent samples of the condition variable and let the empirical expected value, $\overline{X} = X/n$, be an estimate of $p$.

**Error:** To bound error in its estimate, Uncertain⟨T⟩'s runtime computes $\Pr[\overline{X} \in [p-\epsilon, p+\epsilon]] \geq 1-\alpha$. In words, it tests if there is at most an $\alpha$ chance that Uncertain⟨T⟩'s estimate of $p$ is wrong. Otherwise, its estimate of $p$ is within $\epsilon$ of the truth. By changing $\alpha$ and $\epsilon$ a programmer can control false-positives and false-negatives, respectfully, while balancing $n$, the number of samples required to evaluate the conditional at that level of confidence and accuracy. If the upper bound of the confidence interval of $\overline{X}$ is less than $p-\epsilon$, the test returns false. Likewise, if the lower bound of the confidence interval of $\overline{X}$ is greater than $p+\epsilon$, the test returns true.

**Sequential probability ratio test:** To implement this, we build a sequential acceptance plan. Let $H_0 : p+\epsilon$ and $H_A : p-\epsilon$ where $p = 0.5$ by default and can be overloaded by a programmer. Uncertain⟨T⟩ calculates the cumulative log-likelihood ratio for each sample:

$$\Delta_L = k \log(H_A/H_0) + (n-k)\log(H_0/H_A)$$

where $n$ is the number of samples taken thus far and $k$ is the number of successes out of those $n$ trials. If

$$\Delta_L \leq \log(alpha/(1-alpha))$$

then Uncertain⟨T⟩ evaluates the conditional as false while if

$$\Delta_L \geq \log((1-alpha)/alpha)$$

the conditional is true.

This process continues drawing samples (and re-calculating $n$ and $k$) until either (i) a bounded number of samples are drawn or (ii) either of the two above conditions are reached.

**A conservative upper bound:** The true power of Wald's SPRT comes into play when compared against a static upper bound of the number of samples required to gain statistical significance. With $\alpha = 0.05$ and $\epsilon = 0.05$, we use Chernoff bounds to compute a conservative upper bound on the number of samples, $N = 2,666$, which is almost two orders of magnitude more than what we commonly see in our experiments(Chen 2011). Likewise, if we change our confidence slightly (i.e., to $\alpha = 0.01$), the number of samples required jumps to $N = 13,333$.

Given that each sample is a poll, which costs money, dynamically estimating the number of samples with Wald's SPRT test dramatically reduces a poll's cost.

## Experiments

This section explores some important properties of power analysis through a series of experiments. These correspond to real queries we have run on Mechanical Turk using InterPoll. Our focus is on both initial hypothesis testing as well as hypothesis *refinement* and query reformulation, which often takes place as a result of analyzing the data we receive from the crowd.

Note that for these experiments we use U.S.-only workers on Mechanical Turk. We set the reward amount to $0.10 per survey completion. Currently, the reward is independent of the length of the survey, although in the future we plan to experiment with reward setting.

We report on three cases studies: (1) a study of male and female heights; (2) an evaluation of anxiety and depression levels of the population; (3) we consider ten polls previously orchestrated by Intelligence Squared US, a debate and opinion polling program broadcast on NPR.

### Height

We start with a simple query that asks participants to provide their height. To simplify our analysis, we request the height in centimeters, as shown below.

```
var people = GetPeople("Height survey",...);
var height = (from person in people
    select new
    {
        Height = person.PoseQuestion<int>(
        "What is your height, in centimeters?"),
        Gender = person.Gender,
        Ethnicity = person.Ethnicity,
    });
```

A natural hypothesis to test with our data is whether males are taller than females. Consulting Wikipedia suggests that for Americans over 20 years of age the average height is 176.3 cm (5 ft 9 1/2 in) for males and 162.2 cm (5 ft 4 in) for females[1]. We attempt to test whether males are generally taller than females with a statement below

```
var maleHeight = from person in height
    where person.Gender == Gender.MALE
    select person.Height;

var femaleHeight = from person in height
    where person.Gender == Gender.FEMALE
    select person.Height;

if(maleHeight.ToRandomVariable() >
    femaleHeight.ToRandomVariable())
{
    Console.WriteLine("Males are taller than females");
}else{
    Console.WriteLine("Males are not taller than females");
}
```

[1] http://en.wikipedia.org/wiki/Template:Average_height_around_the_world

In this case, the power analysis only needs $N = 29$ samples to test this hypothesis.

Looking through the data we observed some unrealistically-looking values of height, stemming from workers either spamming our survey or not being able to convert their height in feet and inches to centimeters properly. To get rid of such outliers, we augmented the query with a simple filter like so

```
height = height.Where(
    p => p.Height > 145 && p.Height < 220);
```

With this change in place, slightly fewer samples are required by power analysis: $N = 27$, perhaps because spurious height values do not distract analysis from the main trend.

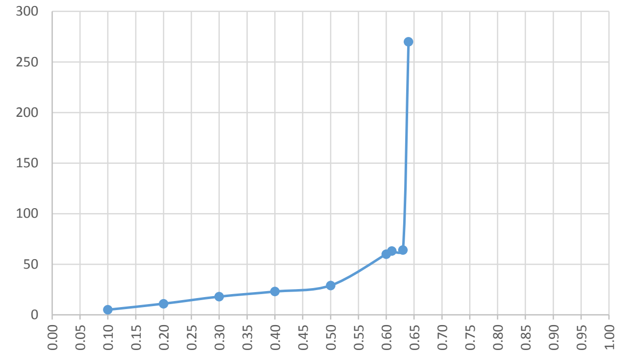
**Changing defaults:** The default setting in InterPoll is a probability value of 0.5 for converting a Bernoulli to a boolean value. We set the confidence value to 0.95 by default. The intuition here is that, if a conditional returns `true`, it is more likely than not to hold. Our next experiment involves changing these defaults to measure their impact on power-analysis. Intuitively, increasing both the probability and the confidence value should increase the number of required samples.

The effects of probability and confidence value changes on power analysis are shown in Figure 4(b) and 4(a), respectively. (Note that we vary both values independently: we increase the probability to .6, etc. while keeping the confidence at .95.) In both cases, there is a cut off point where hitting a certain threshold value leads to rapid growth in the number of samples. The possibility of exponential growth highlights the importance of choosing both probability and confidence parameters carefully. Indeed, for users of the data it may make a very small amount of difference whether the probability is .5 and not .6 and that the confidence is .9 and not .95, yet these small numeric changes of parameters may yield significant power analysis differences, leading to differences in cost.

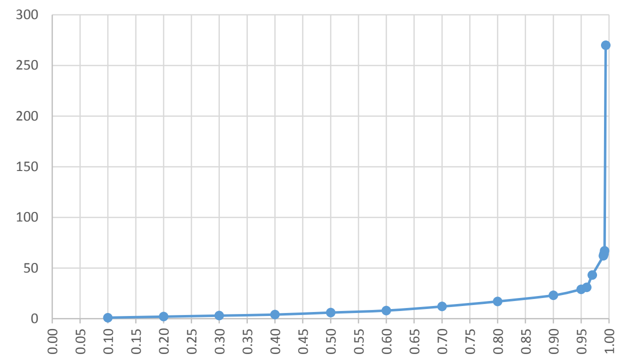
## Does Money Buy Happiness?

There has been an ongoing discussion in the press as to whether poverty is a source of anxiety and various forms of mental illness[2]. To explore this issue further, we decided to collect some anxiety and depression data from the population. Asking people whether they are depressed or anxious may not be the best approach, so instead we use the Hospital Anxiety and Depression Scale (HADS) to calculate two scores for anxiety and depression. Score for each subscale (anxiety and depression) can range from 0–21 with scores categorized as follows: normal (0–7), mild (8–10), moderate (11–14), severe (15–21). These scores come from numerically encoded answers to multiple choice questions.

[2] http://www.medicaldaily.com/poverty-may-be-leading-cause-generalized-anxiety-disorder-not-mental-illness-241457




(a) The effect of probability values on power analysis.




(b) The effect of confidence values on power analysis.

Figure 4: Influence of parameters on Power analysis.

```
var happinessData = from person in people
   select new {
   Consider = person.PoseCodedQuestion(
      "I feel tense or 'wound up'",
      {"Most of the time", 3},
      {"A lot of the time", 2},
      {"From time to time", 1},
      {"Not at all", 0)},
   Enjoy = person.PoseCodedQuestion(
      "I still enjoy the things I used to enjoy",
      {"Definitely as much", 0}
      {"Not quiet so much", 1},
      {"Only a little", 2},
      {"Hardly at all", 3}),
   Awful = person.PoseCodedQuestion(
      "I get a sort of frightened feeling as if " +
      "something awful is about to happen",
      {"Very definitely and quiet badly", 3},
      {"Yes, but not too badly", 2},
      {"A little, but it doesn't worry me", 1},
      {"Not at all", 0}),
   ...
};
```

Note that score calculation based on obtained data — a process often referred to as *coding* — is directly supported by InterPoll queries. From the data above, we compute anxiety and depression scores directly with another LINQ query:

```
var scores = from data in happinessData
    select new
    {
        Anxiety =
            data.Consider + data.Awful + data.Worry +
            data.Sit + data.Frightened + data.Restless +
            data.Panic,
        Depression = data.Enjoy + data.Laugh +
            data.Cheerful + data.Slowed +
            data.LostInterest + data.LookForward +
            data.GoodBook,
        Gender = data.Gender,
        Income = data.Income,
        Education = data.Education,
        Ethnicity = data.Ethnicity,
    };
```

We can then look at anxiety scores of individuals making over \$35,000 per annum (whom we categorize as *rich*) and those making under \$25,000 (whom we categorize as *poor*). Of course, one's definitions of what rich and poor are can vary widely, but these are the subjective choices we made while formulating the queries.

```
var rich = from person in scores
    where
        person.Income == Income.INCOME_35_000_TO_49_999 ||
        person.Income == Income.INCOME_50_000_TO_74_999 ||
        person.Income == Income.INCOME_75_000_AND_OVER
    select person.Anxiety;

var poor = from person in scores
    where
        person.Income == Income.INCOME_1_TO_4_900 ||
        person.Income == Income.INCOME_10_000_TO_14_999 ||
        person.Income == Income.INCOME_15_000_TO_24_999
    select person.Anxiety;
```

The last step involves comparing the anxiety levels of these two population sub-groups.

```
if (rich.ToRandomVariable() > poor.ToRandomVariable())
{
    Console.WriteLine("Rich are more anxious than poor");
}
else {
    Console.WriteLine("Rich are no more anxious than poor");
}
```

Running the code above, power analysis decides that $N = 105$ is the numbers of samples that are needed. The answer to this question is a *No*, that is rich are no more anxious that poor. Moreover, checking the expected value, we discover that for `poor` is is 8.57 and for `rich` it is 8.0. Given the expected value for rich is somewhat lower, it is probably not surprising that we cannot prove rich to be more anxious than poor.

## Intelligence Squared U.S. Debates

Intelligence Squared U.S. is a NRP-broadcast program which organizes Oxford-style debates live from New York City. Intelligence Squared U.S. has presented more than 85 debates on a wide range of often provocative topics, which range from clean energy and the financial crisis, to the situation in the Middle East.

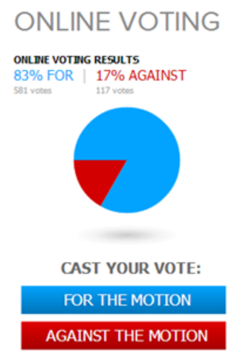



Figure 5: Polling results for the motion "Should we break up the big banks?"

Every debate consists of a motion, with debaters, who are often recognized experts in their fields, arguing for and against the motion. Prior to each debate, Intelligence Squared U.S. organizes an online poll to get a sense of public opinion on the matter being debated. An example of such a pre-debate poll obtained from `http://intelligencesquaredus.org/debates/past-debates/item/906-break-up-the-big-banks` is shown in Figure 5.

Not surprisingly, the program selects contentious, too-close-to-call topics where neither side easily dominates. As such, these debates present an interesting challenge for our power analysis, compared to easy-to-decide issues (such as male-vs-female height).

We have implemented a total of 10 debates obtained directly from `http://intelligencesquaredus.org/debates/past-debates/` site, focusing on topics such as the economy, foreign policy, income inequality, etc. All the debate polls we have implemented follow the same pattern shown in Figure 6, modulo changes to the question text.

Figure 7 shows a summary of our results for each of the debate polls. Alongside the outcome of each poll, we show the power analysis-computed value of $N$. We also show the dollar cost required to obtain the requisite number of samples from the crowd.

Conceptually, it may be instructive to separate the polls into "easy-to-decide", "contentions", and "truly contentions." For instance, `ObesityIsGovernmentBusiness` was the most costly debate of them all, requiring 265 workers to share their attitudes. Our of these, 120 (45%) were yes votes, whereas 145 (55%) said no.

**Slicing and Data Analysis** Next we analyze the data from the debate polls above and highlight the influence of different demographic characteristics on the results. Finding insight in the data is not an easy task, and is one that eludes automation. Much of the time we

```
var question = "Do you believe the rich are taxed enough?";
var frame = (from person in people
    select new
    {
        Outcome = person.PoseQuestion<bool>(
        question),
        Gender = person.Gender,
        Ethnicity = person.Ethnicity,
        Age = person.Age,
        Education = person.Education,
    });
var answer = from person in frame
        select person.Outcome;

if (answer.ToRandomVariable().Pr(.5)) {
    Console.WriteLine("Outcome: {0} : Yes", question);
} else {
    Console.WriteLine("Outcome: {0} : No", question);
}
```

Figure 6: Debate polling code in InterPoll.

would analyze the data we obtained using pivot table tools in Excel to see if interesting patterns emerge.

**Gender:** Figure 8(a) shows the connection between affirmative action attitudes and gender. As the figure shows, women are generally more positive toward affirmative action on campus, whereas men feel that it does more harm than good in larger numbers.

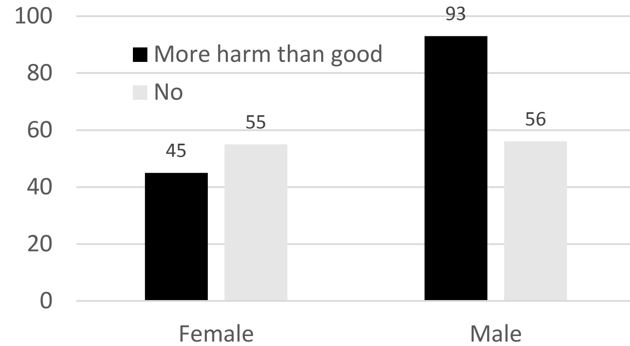
Similarly, when we look at the attitudes toward taxing the rich, females think that rich are not taxed enough in disproportionally higher numbers, as shown in Figure 8(b). While looking at the data makes it clear that for our sample, females do not believe rich are taxed enough, we can test this as a hypothesis by adjusting the query frame to only include women:

```
frame = from person in frame
    where person.Gender == Gender.FEMALE
    select person;
```

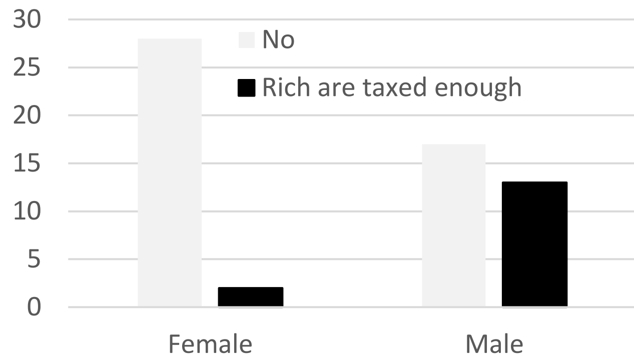
A subsequent test of their preferences for taxation reveals that indeed, women do not believe the rich are taxed high enough; the power analysis requires $N = 51$ samples.

| Task | Outcome | Power | Cost |
|---|---|---|---|
| `MilennialsDontStandAChance` | No | 37 | $3.70 |
| `MinimumWage` | No | 43 | $4.30 |
| `RichAreTaxedEnough` | No | 51 | $5.10 |
| `EndOfLife` | No | 53 | $5.30 |
| `BreakUpTheBigBanks` | Yes | 73 | $7.30 |
| `StrongDollar` | No | 85 | $8.50 |
| `MarginalPower` | No | 89 | $8.90 |
| `GeneticallyEngineeredBabies` | Yes | 135 | $13.50 |
| `AffirmativeActionOnCampus` | Yes | 243 | $24.30 |
| `ObesityIsGovernmentBusiness` | No | 265 | $26.50 |

Figure 7: Ten debates: outcomes, power analysis, and costs.



(a) Affirmative action attitudes by gender.



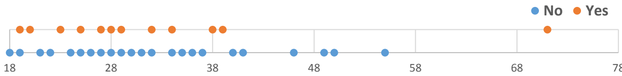
(b) Taxation attitudes attitudes by gender.

Figure 8: Dependency on gender.

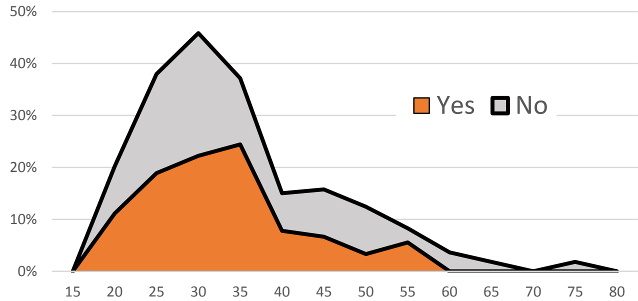
**Age:** We have explored several results to see if they have a strong age-based component. Overall, we find that the dependency on gender is stronger than that on age, but of course, this is highly question-specific. Figure 9a shows the questions to an end-of-life poll (*Should we ration end-of-life care?*). ”Yes” answers are shown in orange and ”No” answers are shown in blue. We see that after a certain age (about 38), all the answers but one are a *No*. Of course, the overall number of answers is too small to have statistical significance, but this is an interesting observation we may be able to test by forming an age-based filter:

```
var younger = from person in frame
    where person.Age < 38 select person.Answer;
var older = from person in frame
    where person.Age >= 38 select person.Answer;
if(younger.ToRandomVariable() >
    older.ToRandomVariable() ) ...
```

Figure 9b shows answers to the question *Do you believe that Russia is a marginal power?* plotted against participant age. Out of 200 people, most participants think that the answer is *No*. To identify age-based differences, we normalize by the number of responders for each curve. We see that in this case, 25–35 year olds disproportionately believe that the answer is a *No*.




(a) End-of-life poll: *Yes* and *No* by age.




(b) Russia is a marginal power: *Yes* and *No* histogrammed by age.

Figure 9: Dependency on age.

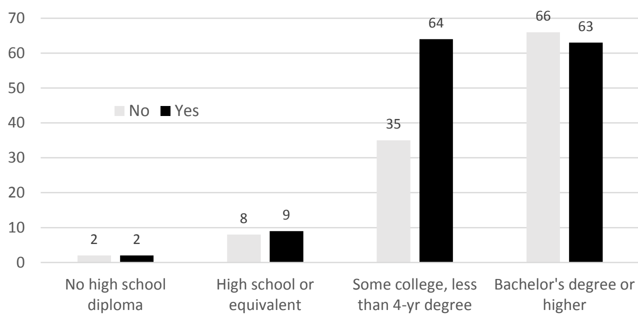



Figure 10: Affirmative action by education.

**Education:** Consider one of the larger polls, `AffirmativeActionOnCampus`, for which we have 243 samples, Figure 10 shows that people who have *incomplete* college education are more supportive of affirmative action than those who have a bachelor's degree (or above).

**Maps:** One of the challenges with obtaining representative polls is the issue of geographic distributions of answers. Figure 11 shows the locations of 250 or so respondents on a map of the US. The plot was constructed based on self-reported ZIP code data. It is useful to perform a casual inspection of the geographic distribution of the data to make sure that, for example, East or West coasts are not overly represented. Our maps *roughly* corresponds to the population density, which is encouraging.

**Convergence Curves** It is instructing to consider the speed of convergence. Figure 12 shows convergence curves for three polls: `MilennialsDontStandAChance`, `MarginalPower`, and `BreakUpBigBanks`. The value is $\Delta_L$ which varies as $N$ grows until it eventually intersects the upper or lower boundary, computed via Wald's


Figure 11: Geographic distribution for the anxiety/depression poll.

SPRT test.

$$upper = \log(\alpha/(1-\alpha)) = 2.94$$
and
$$lower = \log((1-alpha)/\alpha) = -2.94$$

`MilennialsDontStandAChance` terminates fast, after only 37 samples, whereas the other two, `BreakUpBigBanks` and `MarginalPower` require 73 and 89 samples, receptively. Shape of the curve can suggest other termination criteria; if a curve is flat, we possibly can decide to terminate the process of sampling to avoid exceeding a budget. Exploring these ideas is part of future work.

## Related Work

InterPoll brings together several bodies of prior work from fields that are traditionally not considered to be particularly related, as outline below.

### Crowd-Sourcing Systems

There has been a great deal of interest in recent years in building new systems for automating crowd-sourcing tasks.

**Toolkits:** TurKit (Little et al. 2009) is one of the first attempts to automate programming crowd-sourced systems. Much of the focus of TurkIt is the iterative paradigm, where solutions to crowd-sourced tasks are refined and improved by multiple workers sequentially. AutoMan (Barowy et al. 2012) is a programmability approach to combining crowd-based and regular programming tasks, a goal shared with Truong *et al.* (Truong, Dustdar, and Bhattacharya 2012). The focus of AutoMan is reliability, consistency and accuracy of obtained results, as well as task scheduling. Turkomatic (Kulkarni, Can, and Hartmann 2011; 2012) is a system for expression crowd-sourced tasks and designing workflows. CrowdForge is a general purpose framework for accomplishing complex and interdependent tasks using micro-task markets (Kraut 2011). Some of the tasks involve article writing, decision making, and science journalism, which demonstrates the

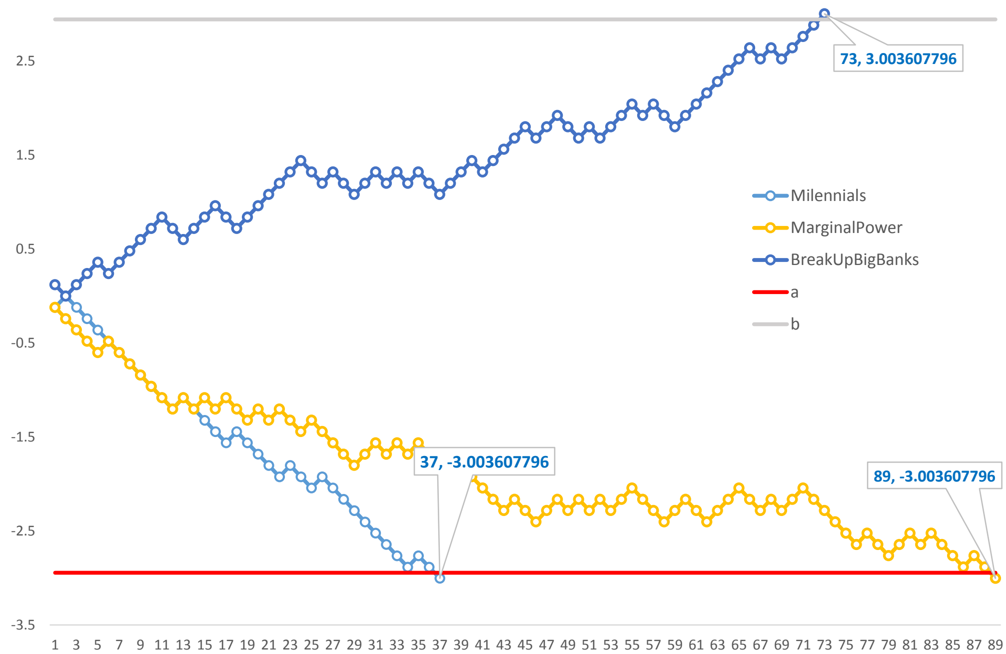



Figure 12: Convergence curves for 3 queries: `MilennialsDontStandAChance`, `MarginalPower`, and `BreakUpBigBanks`.

benefits and limitations of the chosen approach. More recently, oDesk has emerged as a popular marketplace for skilled labor. CrowdWeaver is a system to visually manage complex crowd work (Kittur et al. 2012). The system supports the creation and reuse of crowdsourcing and computational tasks into integrated task flows, manages the flow of data between tasks, etc.

We do not aim to adequately survey the vast quantity of crowd-sourcing-related research out there; the interested reader may consult (Yin et al. 2014). Notably, a great deal of work has focused on matching users with tasks, quality control, decreasing the task latency, etc.

Moreover, we should note that our focus is on subjective *opinion polls* which distinguishes InterPoll work from the majority of crowd-sourcing research which requires giving a solution to a particular task such as deciphering a license plate number in a picture, translating sentences, etc. In InterPoll, we are primarily interested in self-reported opinions of users about themselves and their preferences.

**Some important verticals:** Some crowd-sourcing systems choose to focus on specific verticals. The majority of literature focuses on the four verticals described below. The reader may find it instructive to understand how surveys are used in each domain. In the interests of completeness, we list some of the most pertinent reference below, without summarizing the work.

- social sciences (Ferneyhough 2012; Behrend, Sharek, and Meade 2011; Antin and Shaw 2012; Kraut et al. 2004; Chandler, Mueller, and Paolacci ; Buhrmester and Kwang 2011; Cooper, McCord, and Socha 2011; Gosling et al. 2004; Podsakoff, MacKenzie, and Lee 2003);
- political science and election polls (Stephenson and Crête 2011; Berinsky, Huber, and Lenz 2012; 2010; Sparrow 2006; Behrend, Sharek, and Meade 2011; Keeter 2006; Yeager et al. 2011);
- marketing (HubSpot and SurveyMonkey ; USamp 2013; Evans, Hempstead, and Mathur 2005); and
- health and well-being (Swan 2012a; 2012b; Eysenbach, Eysenbach, and Wyatt 2002; Ramo, Hall, and Prochaska 2011; Wyatt 2000; Behrend, Sharek, and Meade 2011; Berinsky, Huber, and Lenz 2010; Andrews, Nonnecke, and Preece 2003; Schmidt 2010; Curmi and Ferrario 2013).

**Survey sites:** In the last several years, we have seen surveys sites that are crowd-backed. The key distinction between these sites and InterPoll is our focus on optimizations and statistically significant results at the lowest cost. In contrast, survey sites generally are incentivized to encourage the survey-maker to solicit as many participants as possible. At the same time, we draw inspiration from many useful features that the sites described below provide.

SurveyMonkey claims to be the most popular survey building platform (HubSpot and SurveyMonkey ). In recent years, they have added support for data analytics as well as an on-demand crowd. Market research seems to be the niche they are trying to target (SurveyMonkey 2013). SurveyMonkey performs ongoing monitoring of audience quality through comparing the answers they get from their audience to that obtained via daily Gullop telephone polls.

Most survey cites give easy access to non-probability samples of the Internet population, generally without attempting to correct for the inherent population bias. Moreover, while Internet use in the United States is approaching 85% of adults, users tend to be younger, more educated, and have higher incomes (Pew Research Center 2013). Unlike other tools we have found, Google Customer Surveys support re-weighting the survey results to match the deomographics of the Current Population Survey (CPS) (US Census 2010).

Unlike other sites, Google Surveys results have been studied in academic literature. McDonald *et al.* (Mcdonald, Mohebbi, and Slatkin ) compares the responses of a probability based Internet panel, a non-probability based Internet panel, and Google Consumer Surveys against several media consumption and health benchmarks, leading the authors to conclude that despite differences in survey methodology, Consumer Surveys can be used in place of more traditional Internet-based panels without sacrificing accuracy. Keeter *et al.* (Keeter, Christian, and Researcher 2012) present a comparison of results performed at Pew to those obtained via Google Customer Surveys.

Instant.ly and uSamp (USamp 2013) focus primarily on marketing studies and boast an on-demand crowd with very fast turn-around times: some survey are completed in minutes. In addition to rich demographic data, uSamp collects information on the industry in which respondents are employed, their mobile phone type, job title, etc., also allowing to

SocialSci (`http://www.socialsci.com`) is a survey site specializing in social science studies. On top of features present in other platforms, it features dynamic workflows for complex surveys, a vetting system for survey-takers based on credit ratings, many demographic characteristics, deceit pools, IRB assistance, etc. We are not aware of demographic studies of the SocialSci respondent population.

**Statistical Techniques for Surveys:** The issue of statistical validity in the context of surveys has long been of interest to statisticians and social science researchers. Two main schools of thought are prominent: the so-called *frequentist* view and the newer, albeit gaining popularity Bayesian view (Bourguignon and Fournier 2007; Callegaro and DiSogra 2008; Erceg-Hurn and Mirosevich 2008; Hanley, Negassa, and Forrester 2003; Lee and Forthofer 2006; Lee 2006; Lee and Valliant 2009; Loosveldt and Sonck 2008; Lumley 2004; Podsakoff, MacKenzie, and Lee 2003; Schonlau et al. 2009; Valliant and Dever 2011; Vella 1998; Winship and Radbill 1994). InterPoll is frequentist but future work will incorporate a Bayesian perspective to deal with bias correction.

## Future Work

Note that our focus in InterPoll is on subjective opinion polls. As such, the focus on traditional concerns of crowd-sourcing is somewhat diminished. In particular, we do not worry about cheating, consensus, and reward setting nearly as much as some of the other efforts. How to apply some of the previously explored ideas to inherently subjective tasks remains an interesting area of future research.

In this paper we have demonstrates that for a wide variety of polls, a relatively small number of samples is sufficient to arrive at a decision. Yet it is possible that for some to-close-to-call decisions a high cost may be required. The issue of *non-convergence* for power analysis requires more exploration.

Indeed, if we observe that we are unable to make a decision after sampling a large number of instances, we may want to either (1) terminate the poll because we are unlikely to get convergence quickly (2) increase the reward to make faster progress (3) decrease the reward to have the poll running cheaply in the background. AutoMan explores some of these possibilities by changing the cost dynamically to try and speed up consensus among workers (Barowy et al. 2012).

More generally, it is interesting to explore the possibility of statically or partially dynamically predicting the cost of running polls ahead of their (full) execution. This is akin to predicting the cost of long-running database queries, except that latencies are even higher for crowd-based work.

## Conclusions

This paper shows how to formulate a variety of complex surveys and polls from a range of domains, including social sciences, political and marketing polls, and health surveys within InterPoll, thereby demonstrating the expressiveness of the system. Abstractions that are presented in this paper are designed to produce accurate results at a minimum cost.

We have performed three cases studies showing polls of varying orders of complexity. We used ten polls from Intelligence Squared US debates, which have been tried online in recent past to get a rough sense of public sentiment. Relative to existing alternatives like SurveyMonkey, which offers *ad infinitum* sampling, InterPoll explicitly manages accuracy and cost. Our experimental results show that all of the polls we considered can be resolved for under $30, with most costing less than $10.

## References

Andrews, D.; Nonnecke, B.; and Preece, J. 2003. Electronic survey methodology: A case study in reaching hard-to-involve Internet users. *International Journal of . . . .*

Antin, J., and Shaw, A. 2012. Social desirability bias and self-reports of motivation: a study of Amazon Mechanical Turk in the US and India. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.*

Barowy, D.; Curtsinger, C.; Berger, E.; and McGregor, A. 2012. AutoMan: A platform for integrating human-based and digital computation. *Proceedings of the ACM international conference on Object oriented programming systems languages and applications - OOPSLA '12* 639.

Behrend, T. S.; Sharek, D. J.; and Meade, A. W. 2011. The viability of crowdsourcing for survey research. *Behavior research methods*.

Berinsky, A. J. A.; Huber, G. A. G.; and Lenz, G. S. 2010. Using mechanical Turk as a subject recruitment tool for experimental research. *Typescript, Yale* 1–26.

Berinsky, A.; Huber, G.; and Lenz, G. 2012. Evaluating Online Labor Markets for Experimental Research: Amazon.com's Mechanical Turk. *Political Analysis* 20(3):351–368.

Bornholt, J.; Mytkowicz, T.; and Mckinley, K. S. Uncertain<T>: A first-order type for uncertain data. In *International Conference on Architectural Support for Programming Languages and Operating Systems*.

Bourguignon, F., and Fournier, M. 2007. Selection bias corrections based on the multinomial logit model: Monte Carlo comparisons. *of Economic Surveys*.

Buhrmester, M., and Kwang, T. 2011. Amazon's Mechanical Turk: A new source of inexpensive, yet high-quality, data? *on Psychological Science*.

Callegaro, M., and DiSogra, C. 2008. Computing response metrics for online panels. *Public Opinion Quarterly*.

Chandler, J.; Mueller, P.; and Paolacci, G. Methodological concerns and advanced uses of crowdsourcing in psychological research.

Chen, X. 2011. Exact computation of minimum sample size for estimation of binomial parameters. *Journal of Statistical Planning and Inference* 141(8):2622 – 2632.

Cooper, C.; McCord, D. M.; and Socha, A. 2011. Evaluating the college sophomore problem: the case of personality and politics. *The Journal of psychology* 145(1):23–37.

Couper, M. P. 2000. Review: Web surveys: A review of issues and approaches. *The Public Opinion Quarterly* 1–31.

Curmi, F., and Ferrario, M. A. 2013. Online sharing of live biometric data for crowd-support: Ethical issues from system design.

Dillman, D.; Tortora, R.; and Bowker, D. 1998. Principles for constructing Web surveys.

Erceg-Hurn, D., and Mirosevich, V. 2008. Modern robust statistical methods: an easy way to maximize the accuracy and power of your research. *American Psychologist*.

Evans, J.; Hempstead, N.; and Mathur, A. 2005. The value of online surveys. *Internet Research* 15(2):195–219.

Eysenbach, J.; Eysenbach, G.; and Wyatt, J. 2002. Using the Internet for Surveys and Health Research. *Journal of Medical Internet Research* 4(2):e13.

Ferneyhough, E. 2012. Crowdsourcing Anxiety and Attention Research.

Gosling, S.; Vazire, S.; Srivastava, S.; and John, O. 2004. Should we trust web-based studies? A comparative analysis of six preconceptions about Internet questionnaires. *American Psychologist* 59(2):93–104.

Gunn, H. 2002. Web-based surveys: Changing the survey process. *First Monday*.

Hanley, J. A.; Negassa, A.; and Forrester, J. E. 2003. Statistical analysis of correlated data using generalized estimating equations: an orientation. *American journal of*.

HubSpot, and SurveyMonkey. Using online surveys in your marketing. 1–43.

Keeter, S.; Christian, L.; and Researcher, S. 2012. A Comparison of Results from Surveys by the Pew Research Center and Google Consumer Surveys.

Keeter, S. 2006. The impact of cell phone noncoverage bias on polling in the 2004 presidential election. *Public Opinion Quarterly*.

Kittur, A.; Khamkar, S.; André, P.; and Kraut, R. 2012. CrowdWeaver: Visually Managing Complex Crowd Work. *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work - CSCW '12* 1033.

Kraut, R.; Olson, J.; Banaji, M.; Bruckman, A.; Cohen, J.; and Couper, M. 2004. Psychological Research Online: Report of Board of Scientific Affairs' Advisory Group on the Conduct of Research on the Internet. *American Psychologist* 59(2):105–117.

Kraut, R. E. 2011. CrowdForge : Crowdsourcing Complex Work. *UIST* 43–52.

Kulkarni, A. P.; Can, M.; and Hartmann, B. 2011. Turkomatic: automatic recursive task and workflow design for mechanical turk. *CHI'11 Extended Abstracts on Human*.

Kulkarni, A.; Can, M.; and Hartmann, B. 2012. Collaboratively crowdsourcing workflows with turkomatic. *of the ACM 2012 conference on*.

Lee, E. S., and Forthofer, R. N. 2006. *Analyzing complex survey data*.

Lee, S., and Valliant, R. 2009. Estimation for volunteer panel web surveys using propensity score adjustment and calibration adjustment. *Sociological Methods & Research*.

Lee, S. 2006. Propensity score adjustment as a weighting scheme for volunteer panel web surveys. *Journal of official statistics*.

Little, G.; Chilton, L. B.; Goldman, M.; and Miller, R. C. 2009. TurKit: tools for iterative tasks on Mechanical Turk. *Proceedings of UIST* 1–2.

Loosveldt, G., and Sonck, N. 2008. An evaluation of the weighting procedures for an online access panel survey. *Survey Research Methods*.

Lumley, T. 2004. Analysis of complex survey samples. *Journal of Statistical Software*.

Mayo, J. 2008. *LINQ Programming*. McGraw-Hill Osborne Media, 1 edition.

Mcdonald, P.; Mohebbi, M.; and Slatkin, B. Comparing Google Consumer Surveys to Existing Probability and Non-Probability Based Internet Surveys.

Pew Research Center. 2013. Demographics of Internet users.

Podsakoff, P.; MacKenzie, S.; and Lee, J. 2003. Common method biases in behavioral research: a critical review of the literature and recommended remedies. *Journal of Applied Psychology* 88(5):879–903.

Ramo, D.; Hall, S.; and Prochaska, J. 2011. Reliability and validity of self-reported smoking in an anonymous online survey with young adults. *Health Psychology*.

Schmidt, L. 2010. Crowdsourcing for human subjects research. *Proceedings of CrowdConf*.

Schonlau, M.; Soest, A.; Kapteyn, A.; and Couper, M. 2009. Selection bias in Web surveys and the use of propensity scores. *Sociological Methods & Research* 37(3):291–318.

Sparrow, N. 2006. Developing Reliable Online Polls. *International Journal of Market Research* 48(6).

Stephenson, L. B., and Crête, J. 2011. Studying political behavior: A comparison of Internet and telephone surveys. *International Journal of Public Opinion Research*.

SurveyMonkey. 2013. Market Research Survey; Get to know your customer, grow your business.

Swan, M. 2012a. Crowdsourced health research studies: an important emerging complement to clinical trials in the public health research ecosystem. *Journal of Medical Internet Research.*

Swan, M. 2012b. Scaling crowdsourced health studies : the emergence of a new form of contract research organization. 9:223–234.

Truong, H. L.; Dustdar, S.; and Bhattacharya, K. 2012. Programming hybrid services in the cloud. *Service-Oriented Computing* 1–15.

US Census. 2010. Current population survey, October 2010, school enrollment and Internet use supplement file. (October).

USamp. 2013. Panel Book 2013.

Valliant, R., and Dever, J. A. 2011. Estimating propensity adjustments for volunteer Web surveys. *Sociological Methods & Research.*

Vella, F. 1998. Estimating models with sample selection bias: a survey. *Journal of Human Resources.*

Wald, A. 1945. Sequential tests of statistical hypotheses. *The Annals of Mathematical Statistics* 16(2):117–186.

Winship, C., and Radbill, L. 1994. Sampling weights and regression analysis. *Sociological Methods & Research.*

Wyatt, J. 2000. When to use web-based surveys. *Journal of the American Medical Informatics Association.*

Yeager, D.; Krosnick, J.; Chang, L.; and Javitz, H. 2011. Comparing the accuracy of RDD telephone surveys and internet surveys conducted with probability and non-probability samples. *Public Opinion Quarterly.*

Yin, X.; Liu, W.; Wang, Y.; Yang, C.; and Lu, L. 2014. What? How? Where? A Survey of Crowdsourcing. *Frontier and Future Development of.*