# ZaliQL: A SQL-Based Framework for Drawing Causal Inference from Big Data

Babak Salimi Dan Suciu

Department of Computer Science & Engineering
University of Washington
{bsalimi, suciu}@cs.washington.edu

## ABSTRACT

Causal inference from observational data is a subject of active research and development in statistics and computer science. Many toolkits have been developed for this purpose that depends on statistical software. However, these toolkits do not scale to large datasets. In this paper we describe a suite of techniques for expressing causal inference tasks from observational data in SQL. This suite supports the state-of-the-art methods for causal inference and run at scale within a database engine. In addition, we introduce several optimization techniques that significantly speedup causal inference, both in the online and offline setting. We evaluate the quality and performance of our techniques by experiments of real datasets.

## 1. INTRODUCTION

To this day, *randomized experiments* (A/B testing) remain the gold standard for causal inference. However, controlled experiments are not feasible or ethical, for economical or practical reasons [**?**]. Fortunately, *Observational studies* can be used to draw causal inference [**?**, **?**, **?**]. Many fields (computational, physical, and social scientists) want to perform causal inference on big data from observational studies: social networks, biological networks, sensor networks and more. Unfortunately, current software for processing observational data for causal inference does not scale. R, Stata, SAS, and SPSS have packages such as *MatchIt* and *CEM*[**?**, **?**], but they are designed for single-table data and are cumbersome with large datasets. For example, we found performing CEM on a dataset with 5M entries takes up to an hour using Stata, R or SAS.

Additionally, causal analysis is part of a larger pipeline that includes data acquisition, cleaning, and integration. For large datasets, these tasks are better handled by a relational database engine.

For this workshop, we propose a causal inference library for PostgreSQL. This takes the first step towards truly scalable causal inference by modeling it as a data management problem. We demonstrate that causal inference can be approached from this perspective, and that doing so is key to scalability and robustness. To demonstrate the library's capabilities, we will run causal analysis on two datasets.

Example 1. Lalonde Dataset
Our first dataset is from the National Supported Work Demonstration, a U.S. job training program. Corey: [cite lalonde 1986] The purpose of the program (treatment) was to increase participants' income, thus 1978 earnings (re78) is the outcome variable. Lalonde (1986) generated both an experimental and observational dataset which are still used to this day to benchmark observational causal inference methods. This dataset contains less than 1000 entries. Throughout the paper, we will use this dataset as an example to explain observational causal inference techniques at a high level.

Example 2. Flight Weather Merged Dataset
Our second dataset demonstrates the capabilities of bringing causal inference into the DBMS. We acquired flight departure details for all commercial flights within the US from 2000 to 2015 (105M entries) and combined this with historical weather data (35M entries). These are relatively large data sets for causal inference and cannot be handled by existing tools. We will use this dataset in our experimental results section to demonstrate the scalability of our library.

Corey: [I think we can skip the tables of attributes]

Babak: [Corey revisit this background in the light of the Lalonde data]

## 2. BACKGROUND: CAUSALITY INFERENCE IN STATISTICS

This section describes the Neyman-Rubin Causal Model (NRCM), which is the basic causal model in statistics.

*Average Treatment Effect (ATE).* In the NRCM we are given a table $R(T, X, Y(0), Y(1))$ with $N$ rows called *units*, indexed by $i = 1 \dots N$; see Table 1. The binary attribute $T$ is called *treatment assignment* ($T = 1$ means the unit was treated; $T = 0$ means the unit was subjected to control); $X$ is a vector of attributes called *covariates*, unaffected by treatment; and the two attributes $Y(0), Y(1)$ represent *potential outcomes*: $Y(1)$ is the outcome of the unit if it is exposed to the treatment and $Y(0)$ is the outcome when it is exposed to the control. For any attribute $Z$ we write $Z_i$ for the value of the $i$'s unit. The effect caused by the treatment for the

| Unit | $T$ (Treatment) | $X$ (Covariates) | $Y(1)$ (Treated outcome) | $Y(0)$ (Control outcome) | $Y(1)-Y(0)$ (Causal Effect) |
|---|---|---|---|---|---|
| 1 | $T_1$ | $X_1$ | $Y_1(1)$ | $Y_1(0)$ | $Y_1(1)-Y_1(0)$ |
| 2 | $T_2$ | $X_2$ | $Y_2(1)$ | $Y_2(0)$ | $Y_2(1)-Y_2(0)$ |
| ... | | | | | |
| N | $T_N$ | $X_N$ | $Y_N(1)$ | $Y_N(0)$ | $Y_N(1)-Y_N(0)$ |

Table 1: The Neyman-Rubin Causal Model (NRCM).

$i$th unit, simply called the *treatment effect* for the $i$th unit, is defined as $Y_i(1) - Y_i(0)$. The goal of causal analysis is to compute the *average treatment effect (ATE)*:

$$\tau_{ATE} = \mathbb{E}[Y(1) - Y(0)] = \mathbb{E}[Y(1)] - \mathbb{E}[Y(0)] \quad (1)$$

Throughout this paper $\mathbb{E}[Z]$ refers to the expected value of the attribute $Z$ of an individual chosen at random from a large population. The population is unavailable to us, instead we have the database which is typically a random sample of $N$ *units* from that population. Then $\mathbb{E}[Z]$ is estimated by the empirical expected value, $\mathbb{E}[Z] = \sum_i Z_i/N$, where $Z_i$ is the attribute of the $i$'th unit in the database. In this paper we do not address the sampling error problem, but we point out that the precision of the estimator increases with the sample size.

Example 3. Lalonde (Cont.) In the Lalonde dataset, $T$ is true if the subject, participated in the job training program. We are measuring the difference in income between those subjects who participated in the program and those who did not, or $Y_i(1) - Y_i(0)$

Corey: [talk about lalonde instead of flight here]

Somewhat surprisingly, the Neyman-Rubin causal model (NRCM) assumes that *both* $Y_i(1)$ and $Y_i(0)$ are available for each unit $i$. For example, if the treatment $T$ is attending the job training program and the outcome is the subject's income, then the assumption is that we have *both* values of subject's income, as if they attended the training program, $Y_i(1)$, *and* if that same subject did not, $Y_i(0)$. The inclusion of both outcomes, factual and counterfactual, in the data model is considered to be one of the key contributions of the NRCM. Of course, in reality we have only one of these outcomes for each unit, *e.g.*, if a subject attended the job training program we know $Y_i(1)$ but not $Y_i(0)$, and in this case we simply write $Y_i$ to denote $Y_i(T_i)$. This missing value prevents us from computing $\tau_{ATE}$ using Eq.(1), and is called the *fundamental problem of causal inference* [**?**]. Therefore, in order to compute $\tau_{ATE}$, statisticians make further assumptions.

***Randomized Data.*** The strongest is the *independence assumption*, which states that the treatment mechanism is independent of the potential outcomes, i.e., $(Y(1), Y(0)) \perp\!\!\!\perp T$. Then, it holds that $\mathbb{E}[Y(1)] = \mathbb{E}[Y(1)|T = 1]$ and similarly $\mathbb{E}[Y(0)] = \mathbb{E}[Y(0)|T = 0]$ and we have:[1]

$$\tau_{ATE} = \mathbb{E}[Y(1)|T = 1] - \mathbb{E}[Y(0)|T = 0] \quad (2)$$

Each expectation above is easily computed from the data, for example $\mathbb{E}[Y(1)|T = 1] = \sum_{i:T_i=1} Y_i/N_1$ where $N_1$ is the number of units with $T = 1$. The gold standard in causal analysis is a *randomized experiment*, where treatments are assigned randomly to the units to ensure independence; for example, in medical trials each subject is randomly assigned the treatment or a placebo.

***Observational Data.*** In this paper, however, we are interested in causal analysis in *observational data*, where the mechanism used to assign treatments to units is not known, and where independence fails in general. The vast majority of datasets available to analysts today are observational . Here, the statistics literature makes the following weaker assumption [**?**]:

**Strong Ignorability:** Forall $x$ the following hold:
(1) Unconfoundedness $(Y(0), Y(1) \perp\!\!\!\perp T|X = x)$ and
(2) Overlap $0 < \Pr(T = 1|X = x) < 1$

The first part, *unconfoundedness*, means that, if we partition the data by the values of the covariate attributes $X = x$, then, within each group, the treatment assignment and the potential outcomes are independent; then we can estimate $\tau_{ATE}$ by computing Eq. 2 for each value of the covariates $X = x$ (i.e., conditioning on $X$) and averaging. The second part, *overlap* is needed to ensure that the conditional expectations $\mathbb{E}[Y(1)|T = 1, X = x]$ and $\mathbb{E}[Y(0)|T = 0, X = x]$ are well defined. For an illustration of unconfoundedness, suppose we restrict to flights on dates with similar weather and traffic conditions, by a fixed carrier, from a fixed airport, etc; then it is reasonable to assume that $T$ and $Y(0)$ are independent and so are $T$ and $Y(1)$. In order to satisfy unconfoundedness, one must collect a sufficient number of confounding attributes in $X$ in order to ignore any indirect correlations between the treatment and the outcome.

However, once we include a sufficient number of covariate attributes $X$, as we should, the data becomes sparse. Many groups $X = x$ are either empty or have a very small number of subjects. For example, if a group $X = x$ has only treated units, then the overlap condition fails, in other words the conditional expectation $\mathbb{E}[Y(0)|X = x, T = 0]$ is undefined.

In general the strong ignorability assumption is not sufficient to estimate $\tau_{ATE}$ on observational data.

***Perfect Balancing.*** The solution adopted in statistics is to increase the size of the groups, while ensuring that strong ignorability holds *within each group*. In other words, instead of grouping by the values of the covariates $X$, one groups by the values of some function on the covariates $B(X)$. We say that the groups are strongly ignorable if the strong ignorability condition holds within each group:

**Strong Ignorability in Groups:** Forall $b$ the following holds:
(1) Unconfoundedness $(Y(0), Y(1) \perp\!\!\!\perp T|B(X) = b)$ and
(2) Overlap $0 < \Pr(T = 1|B(X) = b) < 1$

Rosenbaum and Rubin [**?**] give an elegant characterization of the functions $B$ that defines strongly ignorable groups, which we review here. We say that the groups are *perfectly balanced*, or that $B$ is a *balancing score* if $X$ and $T$ are independent in each group, *i.e.*$(X \perp\!\!\!\perp T|B(X) = b)$ for all values $b$. Equivalently:

[1] An additional assumption is actually needed, called *Stable Unit Treatment Value Assumptions (SUTVA)*, which states that the outcome on one unit is not affected by the treatment of another unit, and the treatment attribute $T$ is binary. We omit some details.

| Name | $\delta(x_i, x_j)$ = | Comments |
|---|---|---|
| Coarsened distance | 0 if $C(x_i) = C(x_j)$<br>$\infty$ if $C(x_i) \neq C(x_j)$ | Where $C(x)$ is a function that coarsen a vector of continues covariate [?] |
| Propensity score distance (PS) | $\lvert E(x_i) - E(x_j)\rvert$ | where $E(x) = \Pr(T = 1 \lvert X = x)$ is the propensity score [?] |
| Mahalanobis Distance (MD) | $(x_i - x_j)'\Sigma^{-1}(x_i - x_j)$ | where $\Sigma$ = covariance matrix [?] |

Figure 1: Distance Measures used in Matching

**Perfect Balanced Groups:** Within each group $b$, the distribution of the covariate attributes of the treated units is the same as the distribution of the control units:

$$\forall x : \Pr(X = x | T = 1, B(X) = b) = \Pr(X = x | T = 0, B(X) = b) \quad (3)$$

THEOREM 1. *[?, Th.3] If the treatment assignment is strongly ignorable, and $B$ defines perfectly balanced groups, then the treatment assignment is strongly ignorable within each group.*

PROOF. The overlap part of the theorem is trivial, we show unconfoundendess. Abbreviating $Y = (Y(0), Y(1))$, we need to prove: if (a) $(Y \perp\!\!\!\perp T | X = x)$ and (b) $(X \perp\!\!\!\perp T | B(X) = b)$, then[2] $(Y \perp\!\!\!\perp T | B(X) = b)$. (a) implies $\mathbb{E}[T|Y = y, X = x] = \mathbb{E}[T|X = x]$ and also $\mathbb{E}[T|Y = y, X = x, B = b] = \mathbb{E}[T|X = x, B = b]$ since $B$ is a function of $X$; (b) implies $\mathbb{E}[T|X = x] = \mathbb{E}[T|X = x, B = b] = \mathbb{E}[T|B = b]$. Therefore, $\mathbb{E}[T|Y = y, B = b] = \mathbb{E}_x[\mathbb{E}[T|Y = y, X = x, B = b]] = \mathbb{E}_x[\mathbb{E}[T|X = x, B = b]] = \mathbb{E}[T|B = b]$ proving the theorem. □

If the treatment assignment is strongly ignorable within each group, then we can compute $\tau_{ATE}$ by computing the expectations of Eq. 1 in each group, then taking the average (weighted by the group probability):

$$\begin{aligned}\tau_{ATE} &= \mathbb{E}[Y(1) - Y(0)] = \mathbb{E}_b[\mathbb{E}[Y(1) - Y(0)|B(X) = b]] \quad (4)\\ &= \mathbb{E}_b[\mathbb{E}[Y(1)|B(X) = b]] - \mathbb{E}_b[\mathbb{E}[Y(0)|B(X) = b]]\\ &= \mathbb{E}_b[\mathbb{E}[Y(1)|T = 1, B(X) = b]] - \mathbb{E}_b[\mathbb{E}[Y(0)|T = 0, B(X) = b]]\end{aligned}$$

Thus, one approach to compute causal effect in observational data is to use group the items into balanced groups, using a balancing fuction $B$. Then, $\tau_{ATE}$ can be estimated using the formula above, which can be translated into a straightforward SQL query using simple selections, group-by, and aggregates over the relation $R$. This method is called *subclassification* in statistics. The main problem in subclassification is finding a good balancing score $B$. Rosenbaum and Rubin proved that the best balancing score is the function $E(x) = \Pr(T = 1|X = x)$, called *propensity score*. However, in practice the propensity score $E$ is not available directly, instead it must be learned from the data using logistic regression, and this leads to several problems [?]. When no good balancing function can be found, a related method is used, called matching.

***Matching.*** We describe matching following [?]. Consider some balancing score $B(X)$ (for example the propensity score). One way to estimate the quantity in Eq. 4 is as follows. First randomly sample the value $b$, then sample one treated unit, and one control unit with $B(X) = b$. This results in a set of treated units $i_1, i_2, \ldots, i_m$ and a matching set of control units $j_1, j_2, \ldots, j_m$: then the difference of their average outcome $\sum_k (Y_{i_k}(1) - Y_{j_k}(0))/m$ is an unbiased estimator of $\mathbb{E}_b[\mathbb{E}[Y(1) - Y(0)|B(X) = b]]$ and, hence, of $\tau_{ATE}$. Generally, the matching technique computes a subset of units consisting of all treated units and, for each treated unit, a randomly chosen sample of fixed size of control units with the same value of balancing score.

Historically, matching predated subclassification, and can be done even when no good balancing score is available. The idea is to match each treated unit with one or multiple control units with "close" values of the covariate attributes $X$, where closeness is defined using some distance function $\delta(x_i, x_j)$ between the covariate values of two units $i$ and $j$. The most commonly used distance functions are listed in Fig. 1. [3] The efficacy of a matching method is evaluated by measuring *degree of imbalance* i.e., the differences between the distribution of covariates in two groups in the matched subset. Since there is no generic metric to compare two distributions, measures such as mean, skewness, quantile and multivariate histogram are used for this purpose. A rule of thumb is to evaluate different distance metrics and matching methods until a well-balance matched subset with a reasonable size obtained.

***Summary.*** The goal of causal analysis is to compute $\tau_{ATE}$ (Eq. 1) and the main challenge is that each record misses one of the outcomes, $Y(1)$ or $Y(0)$. A precondition to overcome is to ensure strong ignorability, by collecting sufficiently many covariate attributes $X$. For the modern data analyst this often means integrating the data with many other data sources, to have as much information available as possible about each unit. One caveat is that one should not include attributes that are themselves affected by the treatment; the principled method for choosing the covariates is based on graphical models [?]. Once the data is properly prepared, the main challenge in causal analysis is *matching* data records such as to ensure that the distribution of the covariates attributes of the treated and untreated units are as close as possible (Eq.(3)). This will be the focus of the rest of our paper. Once matching is performed, $\tau_{ATE}$ can be computed using Eq.(4). Thus, the main computational challenge in causal analysis is the matching phase, and this paper describes scalable techniques for performing matching in a relational database system.

Babak: [introduce ZaliQL here]

## 3. BASIC TECHNIQUES

In this section we review the matching and subclassification techniques used in causal inference and propose several relational encodings, discussing their pros and cons. Historically, matching was introduced before subclassification, so

[2] This is precisely the Decomposition Axiom in graphoids [?, the. 1]; see [?] for a discussion.

[3] The distance functions in Table 1 are *semi or pseudo-metrics*. That is they are symmetric; they satisfy triangle inequality and $x_i = x_j$ implies $\delta(x_i, x_j) = 0$, but the converse does not hold.

```
CREATE VIEW NNMWR AS
SELECT *
   FROM R AS control,R AS treated
WHERE control.T=0  AND treated.T=1
  AND δ(treated.X,control.X) < caliper
  AND (SELECT count(*)
     FROM R AS z
     WHERE z.T=0
       AND δ(treated.X,z.X) < δ(treated.X,control.X) ≤ k)
```

**(a) Anti-join based**

```
CREATE VIEW NNMWR AS
WITH potential_matches AS
  (SELECT treated.ID AS tID,control.ID AS cID,
        δ(treated.X,control.X)  AS distance
   FROM R AS control, R AS treated
   WHERE control.T=0 AND treated.T=1
     AND δ(treated.X,control.X) < caliper),
     ranked_potential_matches AS
  (SELECT *, ROW_NUMBER() OVER (PARTITION BY tID
                       ORDER BY distance) AS order
   FROM potential_matches)
SELECT *
FROM ranked_potential_matches
WHERE order ≤ k;
```

**(b) Window function based**

Figure 2: **SQL implementation of NNMWR.**

we present them in this order. Subclassification is the dominant technique in use today: we will discuss optimizations for subclassification in the next section.

We consider a single relation $R(ID, T, X, Y)$, where $ID$ is an integer-valued primary-key, $T$ and $X$ respectively denote the treatment and covariate attributes as described in the NRCM (cf. Section 2), and $Y$ represent the available outcome, i.e., $Y = Y(z)$ for iff $T = z$. For each matching method, we define a view over $R$ such that materializing the extension of the view over any instance of $R$ computes a corresponding matched subset of the instance.

```
CREATE VIEW NNMNR
AS WITH potential_matches AS
  (SELECT treated.ID AS tID, control.ID AS cID,
          δ(treated.X,control.X)  AS distance
   FROM R AS control, R AS treated
   WHERE control.T=0 AND treated.T=1
     AND δ(treated.X,control.X) < caliper)),
         ordered_potential_matches AS
  (SELECT *, ROW_NUMBER() over (ORDER BY distance) AS order
   FROM potential_matches)
SELECT *
FROM ordered_potential_matches AS rp
WHERE NOT EXISTS
    (SELECT *
     FROM ordered_potential_matches AS z
     WHERE z.order < rp.order AND z.cID=rp.cID)
  AND (SELECT count(*)
     FROM ordered_potential_matches AS rp
     WHERE z.order < rp.order AND z.tID=rp.tID) ≤ k;
```

Figure 3: **SQL implementation of NNMNR**

Babak: [ We should decide whether we want to include the NNM stuff]

## 3.1 Nearest Neighbor Matching

The most common matching method is that of $k:1$ nearest neighbor matching (NNM) [**?**, **?**, **?**]. This method selects the $k$ nearest control matches for each treated unit and can be done with or without replacement; we denote them respectively by NNMWR and NNMNR. In the former case, a control unit can be used more than once as a match, while in the latter case it is considered only once. Matching with replacement can often decrease bias because controls that look similar to the treated units can be used multiple times. However, since control units are no longer independent, complex inference is required to estimate the causal effect [**?**]. In practice, matching is usually performed without replacement. Notice that NNM faces the risk of bad matches if the closest neighbor is far away. This issue can be resolved by imposing a tolerance level on the maximum distance, known as the *caliper* (see e.g., [**?**]).

**NNM With Replacement** We propose two alternative ways for computing NNMWR in SQL, shown in Figure 2. In Figure 2(a), each treated unit is joined with $k$ closest control units that are closer than the caliper. In this solution, nearest control units are identified by means of an anti-join. In Figure 2(b), all potential matches and their distances are identified by joining the treated with the control units that are closer than the caliper. Then, this set is sorted into ascending order of distances. In addition, the order of each row in the sorted set is identified using the window function `ROW_NUMBER`. Finally, all units with the order of less than or equal to $k$ are selected as the matched units.

The ani-join based statement requires a three-way join. The window function based solution has a quadratic complexity. It requires a nested-loop to perform a spatial-join and a window aggregate to impose minimality. Note that window functions are typically implemented in DBMS using a sort algorithm, and even more efficient algorithms have been recently proposed [**?**].

**NNM Without Replacement** This method aims to minimize the average absolute distance between matched units and can performed in either greedy or optimal manner. The latter is called *optimal matching* [**?**]. Before we describe our proposed SQL implementation for NNMWR, we prove that optimal matching is not expressible in SQL:

this justifies focusing on approximate matches. For our inexpressibility result, notice that in the special case when $k = 1$ NNMWR is the *weighted bipartite graph matching problem (WBGM)*, which is defined as follows: given a bipartite graph $G = (V, E)$ and a weight function $w : E \to \mathbb{R}_{>0}$, find a set of vertex-disjoint edges $M \subseteq E$ such that $M$ minimise the total weight $w(M) = \sum_{e \in M} w(e)$. The exact complexity of this problem is unknown (see, e.g. [**?**]), however we prove a NLOGSPACE lower bound:

PROPOSITION 1. *Computing maximum weight matching for weighted bipartite graphs is hard for* NLOGSPACE.

PROOF. The following *Graph Reachability Problem* is known to be NLOGSPACE complete: given a directed graph $G(V, E)$ and two nodes $s, t$, check if there exists a path from $s$ to $t$. We prove a reduction from graph reachability to the *bipartite perfect matching problem* which is a special case of optimal WBGM. For that we construct the graph $G'$ with $V = V \cup V'$ where, $V'$ is a copy of $V$ with primed labels and $E' = \{(x, x') | \forall x \in V - \{s, t\}\} \cup \{(x, y') | \forall (x, y) \in E\} \cup \{(t, s')\}$. Notice that the subset $\{(x, x') \mid x \in V\} \subseteq E$ is almost a perfect matching, except that it misses the nodes $s, t'$. We prove: there exists a path from $s$ to $t$ in $G$ iff $G'$ has a perfect matching. First assume $P = s, x_1, x_2, \ldots, x_m, t$ is a path in $G$. Then the following forms a perfect matching in $G'$: $M = \{(s, x'_1), (x_1, x'_2), \ldots, (x_m, t'), (t, s')\} \cup \{(y, y') \mid y \notin \{s, x_1, \ldots, x_m, t\}\}$. Conversely, assume $G'$ has a perfect matching. Write $f : V \to V'$ the corresponding bijection, i.e. every $x$ is matched to $y' = f(x)$. Denoting the nodes in $V$ as $V = \{x_1, \ldots, x_n\}$, we construct inductively the following sequence: $x'_{i_1} = f(s)$, $x'_{i_2} = f(x_{i_3})$, $\ldots$, $x'_{i_{k+1}} = f(x_{i_k})$. Then $i_1, i_2, \ldots$ are distinct (since $f$ is a matching), hence this sequence must eventually reach $t'$: $t' = f(x_{i_m})$. Then $s, x_{i_1}, x_{i_2}, \ldots, x_{i_m}, t$ forms a path from $s$ to $t$ in $G$. This completes the proof. □

The proposition implies that optimal matching is not expressible in SQL without the use of recursion. Optimal matching can be solved in PTIME using, for example, the *Hungarian* algorithm, which, in theory, could be expressed using recursion in SQL. However, optimal matching is rarely used in practice and, in fact, it is known that it does not in general perform any better than the greedy NNM (discussed next) in terms of reducing degree of covariate imbalance [**?**]. For that reason, we did not implement optimal matching in our system.

1:1 NNMWR can be approximated with a simple greedy algorithm that sorts all edges of the underlying graph in ascending order of weights and iterates through this sorted list, marking edges as "matched" while maintaining the one-to-one invariant. Figure 3 adopts this greedy algorithm to express $1 : k$ NNMWR in SQL. This algorithm is very similar to that of NNMWR in Figure 2(b), with the main difference that in the matching step it imposes the restriction that a control unit is matched with a treated unit only if it is not not already matched with another treated with a lower order. This solution also has a quadratic complexity.

**Choosing the distance function** We briefly discuss now the choice of the distance function $\delta$ in NNM (see Fig. 1). The propensity score distance is by far the most prominent metric in NNM. However, it has been the subject of some recent criticisms [**?**]. It has been shown that, unlike other matching methods, in propensity score matching the imbalance reduction is only guaranteed across the spectrum of all samples. In observational settings, we typically have only one sample, so other matching methods dominate propensity score matching [**?**]. An alternative is to use the mahalanobis distance. This has been shown to exhibit some odd behavior when covariates are not normally distributed, when there are relatively large number of covariates, or there are dichotomous covariates [**?**]. Therefore, this method has a limited practical applicability.

We should mention that there is huge literature in the database community on finding the nearest neighbor. In fact this type of queries are subject of an active research and development efforts in the context of spatial-databases (see, e.g., [**?**]). Our work is different from these efforts in that: 1) much of the work in this area has focused on finding sublinear algorithm for identifying nearest neighbors of a single data item (e.g., by using spatial-index). In contrast, in our setting we need to find all nearest neighbors, which is by necessity quadratic; 2) these works resulted in specialized algorithm, implemented in general purposed languages. In contrast, we focus on finding a representation in SQL, in order to integrate causal analysis with other data analytic operations.

```
CREATE VIEW SUBC AS
(WITH tmp0 AS
  (SELECT *. ntile(n) over w subclass,
   FROM R window w AS (ORDER BY ps))
SELECT ID, T, X, Y, subclass,
           max(T) over w maxT, min(T) over w minT
FROM tmp0  window w AS (PARTITION BY BY subclass)
WHERE maxT!=minT)
```

Figure 4: **SQL implementation of subclassification based on the propensity score.**

## 3.2 Subclassification

It is easy to see that NNM does not necessarily use all the data, meaning that many control units despite being in the range of a treatment unit are discarded. In subclassification, the aim is to form subclasses for which, the distribution of covariates for the treated and control groups are as similar as possible. It is shown that using just five subclasses based on univariate continues covariates or propensity score removes over 90% of covariates imbalance [**?**, **?**].

**Subclassification based on the propensity score** We describe the SQL implementation of subclassification based on $n$ quintiles of the propensity score in Figure 4. We assumed that $R$ includes another attribute $ps$ for the propensity score of each unit; the value of $ps$ needs to be learned from the data, using logistic regression [**?**]. The SQL query seeks to partition the units into five subclasses with propensity scores as equal as possible using the window function `ntile` and ensure the overlap within each subclass. This solution has the order of $nlog(n)$ if the window function computed using a sort algorithm.

**Coarsening Exact Matching (CEM)** This method as proposed recently in [**?**], is a particular form of subclassification in which the vector of covariates $X$ is coarsened according to a set of user-defined cutpoints or any automatic discretization algorithm. Then all units with similar coarsened covariates values are placed in unique subclasses. All subclasses with at least one treated and one control unit are retained and the rest of units are discarded. Intuitively, this is a group-by operation, followed by eliminating all groups that have no treated, or no control unit.

For each attribute $x_i \in X$, we assume a set of cutpoints

```
CREATE VIEW R^c AS
(SELECT *, (CASE WHEN x_1 < c_1   THEN 1 ...
                 WHEN x_1 > c_(k_1-1) THEN k_1) AS cx_1,
            . . .
           (CASE WHEN x_n < c_n   THEN 1 ...
                 WHEN x_n > c_(k_n-1) THEN k_n) AS cx_n
from R);
```

(a)Coarsening wrt. a set of prespecified cutpoints

```
CREATE VIEW CEM AS
SELECT ID, T, X, Y, subclass
FROM
  (SELECT *,
          max(ID) OVER w AS subclass, max(T) OVER w AS minT,
          min(T) OVER w AS maxT
   FROM R^c
   WINDOW w (PARTITION BY X))
WHERE mint!=maxt
```

(b)Window function based

```
CREATE VIEW CEM AS
WITH subclasses AS
  (SELECT *,
          max(ID) OVER w subclass, max(T) OVER w AS minT,
          min(T) OVER w AS maxT
   FROM R^c
   Group by X)
SELECT ID, T, X, Y, subclass
FROM subclasses,R^c
WHERE subclasses.X=R^c.X  AND  minT!=maxT
```

(c) Group-by based

Figure 5: **SQL implementation of CEM.**

$c_i = \{c_1 \dots c_{(k_i-1)}\}$ is given, which can be used to coarsen $x_i$ into $k_i$ buckets. The view $R^c$, shown in Figure 5(a), defines extra attributes $\mathcal{X} = \{cx_1 \dots cx_n\}$, where $cx_i$ is the coarsened version of $x_i$. Two alternative SQL implementations of CEM are represented in Figure 5(b) and (c). The central idea in both implementations is to partition the units based on the coarsened covariates and discard those partitions that do not enjoy the overlap assumption. Corey: [assumption or condition here?] Note that the maximum of unit $ID$s in each partition is computed and used as its unique identifier. The window function based solution has the order of $nlog(n)$, if the window aggregate computed using a sort algorithm. The group-by based solution can becomes linear if the join is performed by a hash-join.

Several benefits of CEM has been proposed in [**?**]. For instance, unlike other approaches, the degree of imbalance is bounded by the user (through choosing proper cut-points for covariates coarsening), therefore the laborious process of matching and checking for balance is no longer needed. The remainder of this paper focuses on optimization techniques to speed up CEM.

Babak: [replace the optimization techniques with a formal result obtained in the series of meetings we had we Johannes]

Babak: [rewrite the optimization technique]

# 4. ADVANCED TECHNIQUES

## 4.1 Multiple Levels of Treatment

Corey: [this is not as rigorous of an explanation as Lopez and Gutman, but I think it is sufficient?] It is not always sufficient to have binary treatments. For example, flight carrier is categorical with many more than 2 possible values. Let $T = \{t_1, t_2, ..., t_z\}$ be the possible treatment values, with $Y = \{Y(t_1), Y(t_2), ..., Y(t_z)\}$ the set of possible outcomes. Much like the ANOVA test in statistics, we look at the entire collection of treatments $T$ to see if there is some effect in $Y$.

ZaliQL handles this feature with a simple GROUP BY HAVING clause. The analyst provides the number of distinct treatment levels and ZaliQL groups subjects if they are exposed to all levels of the treatment.

**Algorithm 1** Binary Treatments

```
GROUP BY CoarsendCovariates
HAVING min(Treatment) != max(Treatment)
```

**Algorithm 2** Non-Binary Treatments

```
GROUP BY CoarsendCovariates
HAVING count(distinct Treatment) = numTreatmentLevels
```

## 4.2 Multiway CEM

Real world data is scattered among many tables. For example, our flight weather merged dataset consists of tables, such as Airport, Carrier, Flight, Weather, etc. Existing toolkits require the analyst to merge all of this data into one table in order to do CEM. ZaliQL, however, combines this sets with CEM as an optimization. The CEM algorithm prunes groups which do not enjoy the overlap condition. That is, groups that share covariates $X = x$ must have both treated and control subjects. Subjects are pruned when this overlap is not met. When we prune down to the base relation, we see runtime improvements because significantly less rows are joined.

For example, $\textsc{Cem}(R \bowtie S)$ is equivalent to $\textsc{Cem}(\textsc{Cem}(R) \bowtie S)$. To see this, note that all subclasses discarded by performing CEM on $R$ do not satisfy the overlap assumption. It is clear that joining these subclasses with $S$, forms new subclasses that still fail to satisfy the overlap assumption and must be discarded. In the following, we formally state this property.

Let $D$ be a standard relational schema with $k$ relations $R_1 \dots R_k$, for some constant $k \geq 1$. The relation $R_i$ has the following attributes: a primary-key $ID_i$; a foreign-key $FID_i$; a vector of observed attributes $A_i$. Without loss of generality, assume the treatment variable, $T$, is in relation $R_1$. Let $\mathcal{X}_{R_i} \subseteq A_i$ be a vector of coarsened covariates from the relation $R_i$ that is associated with $T$. Further, assume relations are joined in the increasing order of indices.

PROPOSITION 2. *Given an instance of $D$, it holds that:* $\textsc{Cem}(R_1 \bowtie \dots \bowtie R_k) = \textsc{Cem}(\dots \textsc{Cem}(\textsc{Cem}(R_1) \bowtie R_2) \dots \bowtie R_k)$.

Proposition 2 shows that CEM can be pushed to normalized databases. In the worst case, the cost of pushing CEM can be $k-2$ times higher than performing CEM on the integrated table. This happens when relations have a one-to-one relationship and CEM retains all the input data. However, in practice the relations typically have a many-to-one relationship. Moreover, the size of the matched subset is much smaller than the input database. In the FLIGHTDELAY example, each row in the weather dataset is associated with many rows in the flight dataset. In addition, as we see in Section 5.2.1, the size of the matched data is much smaller than the input data. In such settings, pushing CEM down to the base relations can significantly reduce its cost.

```
CREATE VIEW P_S AS
WITH tmp0 AS
  (SELECT *,
          max(ID) OVER w AS supersubclass,
          max(T_1) OVER w AS maxT_1,..., maxT(T_k') OVER w AS maxT_k',
          min(T_1) OVER w AS minT_1,..., minT(T_k') OVER w AS minT_k'
   FROM R^e
   WINDOW w (PARTITION BY X'))
SELECT ID, X, Y , supersubclass
FROM tmp0
WHERE max(T_1)!=max(T_1) or ... or  max(T_k')!=max(T_k')
```

(a)Covariates factoring.

```
CREATE VIEW mCEM_{T_i} AS
WITH tmp0 AS
  (SELECT *,
          max(ID) OVER w    subclass,
          max(T_i)  OVER w AS minT,
          max(T_i)  OVER w AS maxT
   FROM P_{T_i}
   WINDOW w (PARTITION BY supersubclass,  X_{T_i} \ X' ))
SELECT ID,T_i,X_{T_i}, Y, subclass
FROM tmp0
WHERE minT!=maxT
```

(b) Modified CEM

Figure 6: CEM based on covariate factoring.

## 4.3 Multiple Binary Treatments

Matching methods are typically developed for estimating the causal effect of a single treatment on an outcome. However, in practice one needs to explore and quantify the causal effect of multiple treatments. For instance, in the FLIGHTDELAY example, the objective is to quantify and compare the causal effect of different weather types on flight departure delays.

This section introduces online and offline techniques to speed up the computation of CEM for multiple treatments. In the sequel, we consider the relational schema consists of a single relation $R^e(ID, \mathcal{T}, \mathcal{X}, Y)$ (extends that of $R^c$ (cf. Section 3.2) to account for multiple treatments), where $\mathcal{T} = T_1, \ldots, T_k$ is a vector of $k$ binary treatments, each of which has a vector of coarsened covariates $\mathcal{X}_{T_i}$, with $\mathcal{X} = \bigcup_{i=1\ldots k} \mathcal{X}_{T_i}$. Now the view $R^e_{T_i}(ID, T_i, \mathcal{X}_{T_i}, Y)$ over $R^e$ has the same schema as $R^c$ (cf. Section 3.2).

### 4.3.1 (online) Covariate Factoring

A key observation for reducing the overall cost of performing CEM for multiple treatments is that many covariates are shared between different treatments. For instance, flights carrier, origin airport, traffic and many weather attributes are shared between the treatments Thunder and LowVisibility. The central idea in *covariate factoring* is to pre-process the input data wrt. the shared covariates between treatments and uses the result to perform CEM for each individual treatment. This significantly reduces the overall cost of CEM for all treatments, if covariate factoring prunes a considerable portion of the input data.

Let $S \subseteq \mathcal{T}$ be a subset of treatments with $\mathcal{X}' = \bigcap_{T_i \in S} \mathcal{X}_{T_i} \neq \emptyset$. Without loss of generality assume $S = \{T_1 \ldots T_{k'}\}$. Consider the view $P_S$ over $R^e$ as shown in Figure 6(a). Essentially, $P_S$ describes CEM wrt. the disjunction of treatments in $S$ and the shared covariates between them. Figure 6(b) defines the view $\text{MCEM}_{T_i}$ over $P_S$ that describes a modified version of CEM for $T_i$, based on the covariate factoring.

PROPOSITION 3. *Given an instance of $R^e$ and any subset of treatments $S \subseteq \mathcal{T}$ with $\bigcap_{T_i \in S} \mathcal{X}_{T_i} \neq \emptyset$, and for any $T_i \in S$, it holds that $\text{MCEM}_{T_i}(R^e) = \text{CEM}(R^e_{T_i})$.*

PROOF. We sketch the proof for $S = \{T_1, T_2\}$. Covariate factoring on $S$, discard subclasses obtained from group-by on $\mathcal{X}_{T_1} \cap \mathcal{X}_{T_2}$ that have no overlap wrt. both $T_1$ and $T_2$. It is clear that group-by $\mathcal{X}_{T_1}$ is more fine-grained than group-by on $\mathcal{X}_{T_1} \cap \mathcal{X}_{T_2}$, thus, subclasses with no overlap in the latter, form new subclasses in the former that still fail the overlap wrt. both of $T_1$ and $T_2$. □

Proposition 3 shows that CEM for multiple treatments can be performed by covariate factoring. Next we develop a heuristic algorithm that takes advantage of this property to speed up CEM. Before we proceed, we state two observations that lead us to the algorithm.

First, the total cost of performing CEM independently for $k$ treatments is a function of the size of the input database. However, the cost of performing the same task using covariate factoring is a function of the size of the result of covariate factoring. But, the cost of covariates factoring depends on the size of the input. Thus, partitioning the treatments into a few set of groups, which results in pruning a huge portion of the input database, reduces the overall cost of CEM.

Second, we observe that the correlation between the treatments plays a crucial role in the efficacy of covariate factoring. The correlation between two treatments $T$ and $T'$ can be measured by the *phi coefficient* between them, denoted by $\phi$. Consider the following table

| | $T = 1$ | $T = 0$ | Total |
|---|---|---|---|
| $T' = 1$ | $n_{11}$ | $n_{10}$ | $n_{1\bullet}$ |
| $T' = 0$ | $n_{01}$ | $n_{00}$ | $n_{0\bullet}$ |
| Total | $n_{\bullet 1}$ | $n_{\bullet 0}$ | $n$ |

where $n_{11}$, $n_{10}$, $n_{01}$, $n_{00}$, are non-negative counts of number of observations that sum to $n$, the total number of observations. The phi coefficient between $T$ and $T'$ is given by $\phi(T, T') = \frac{n_{11}n_{00} - n_{10}n_{01}}{\sqrt{n_{1\bullet}n_{0\bullet}n_{\bullet 0}n_{\bullet 1}}}$. A phi coefficient of 0 indicates independence, while 1 and -1 indicates complete dependence between the variables (most observations falls off the diagonal). Suppose $T_1$ and $T_2$ are highly correlated, i.e., $|\phi(T, T')| \simeq 1$ and share the covariates $X$. Further assume CEM wrt. $T$ and $X$, prunes 50% of the input data. Then, covariates factoring for $T_1$ and $T_2$ prunes almost 50% of the input data. This is because, subclasses discarded by CEM on $T$, are very likely to be discarded by CEM wrt. the disjunction of $T$ and $T'$ (because there are highly correlated).

Algorithm 3, was developed based on these observations. Section 5.2.4 shows that covariates factoring based on this algorithm significantly reduces the over all cost of CEM wrt. multiple treatments in the FLIGHTDELAY example.

**Algorithm 3** Covariate Factoring

1. Let $T_1 \ldots T_k$ be a set of treatments, and $\mathcal{X}_{T_i}$ be a vector of covariates associated to $T_i$.
2. Construct a correlation matrix, $\mathcal{M}$, between the treatments such that the $[i, j]$ entry in $\mathcal{M}$ contains $\phi(T_i, T_j)$.
3. Given a partition of the treatments into $n$ groups, $S_1 \ldots S_n$, such that $|S_k| \geq 2$ and $\bigcap_{T_i \in S_k} \mathcal{X}_{T_i} \neq \emptyset$, compute the normalized pairwise correlations in $s_k$ as $\mathcal{C}_{S_k} = \frac{\sum_{(T_i, T_j) \in S_k} |\mathcal{M}[i,j]|}{|S_k|}$.
5: Perform covariate factoring for groups obtained by the partition that maximises $\sum_{k=1\ldots n} \mathcal{C}_{S_k}$.

#### 4.3.2 (online) Data-Cube Aggregation

Given a set of observed attributes $X$ and an outcome, all attributes can be subjected to a causal analysis. For example, all weather attributes can be dichotomized and form a treatment. In addition, one may define other treatments, formed by conjunction of such dichotomized attributes. For instance, the conjunction of "snow" and "high-wind-speed" could become the "snowstorm" treatment. Note that causal effect is not subadditive [**?**]. Thus, quantifying the causal effect of the conjunction of $T_1$ and $T_2$ requires an independent analysis on the treatment $T = T_1 \wedge T_2$ wrt. the covariates $X_T = X_{T_1} \cup X_{T_2}$.

In principle, one might be interested in exploring and quantifying the casual effect of $k = 2^{|X|}$ treatments. In this setting, to estimate $\tau_{ATE}$ for all possible treatments, a matching method must be performed wrt. all possible subsets of $X$, each of which is associated to one treatment. We argue that, in such cases, CEM for all treatments can be performed efficiently using the existing DBMS systems that support data-cube operations.

Recall that CEM for an individual treatment is a group-by operation (cf. Figure 5(b)). Thus, CEM for all treatments requires computing some aggregates on the data-cube ($\mathcal{X}$). Now the established optimization techniques to compute data-cubes efficiently can be adopted, e.g., for computing a group-by, we pick the smallest of the previously materialized groups from which it is possible to compute the group-by. In Section 5.2.4, we apply this idea to the FlightDelay example and show that it significantly reduces the overall cost of CEM for multiple treatments.

#### 4.3.3 (offline) Databases Preparation for Causal Inference on Sub-populations

So far, we considered causal inference as an online analysis which seeks to explore the effect of multiple treatments on an outcome, over a population. In practice, however, one needs to explore and quantify the causal effect of multiple treatments over various sub-populations. For instance, what is the causal effect of low-visibility on departure delay in San Francisco International Airport (SFO)? what is the effect of thunder at all airports in the state of Washington since 2010? such queries can be addressed by performing CEM on the entire data and selecting the relevant part of the obtained matched subset to the query.

Thus, the cost of performing CEM wrt. all possible treatments can be amortized over several causal queries. Therefore, we can prepare the database offline and pre-compute the matched subsets wrt. all possible treatments to answer online causal queries efficiently. This could be impractical for high dimensional data since the number of possible treatments can be exponential in the number of attributes (cf. Section 4.3.2). Alternatively, we propose Algorithm 3, which employs the covariate factoring and data-cube techniques to prepare the database so that CEM based on any subset of the database can be computed efficiently.

**Algorithm 4** Database Preparation

1: Let $T_1 \dots T_k$ be a set of treatments, and $\mathcal{X}_{T_i}$ be a vector of covariates associated to $T_i$.
2: Apply Algorithm 3 to partition the treatments into $S_1 \dots S_k$ with $\mathcal{X}_{S_i} = \bigcup_{T_j \in S_i} \mathcal{X}_{T_j}$ and $\mathcal{X}'_{S_i} = \bigcap_{T_j \in S_i} \mathcal{X}_{T_j}$.
3: Partially materialize $\mathcal{C}$, the cube on $\mathcal{X}_1 \dots \mathcal{X}_k$ to answer group-by queries for each $\mathcal{X}'_{S_i}$.
4: For each group $S_i$, perform covariate factoring using $\mathcal{C}$ and materialize $P_{S_i}$.
5: For each $P_{S_i}$, partially materialize $\mathcal{C}_i$, the cube on $\mathcal{X}_{T_i}$, so that CEM for each $T \in g_i$ can be computed using $\mathcal{C}_i$.

# 5. EXPERIMENTAL RESULTS

We have implemented the basic techniques in Sec. 3 and the optimizations in Sec. 4 in a system called ZaliQL. This section, presents experiments that evaluate the feasibility and efficacy of ZaliQL. We addressed the following questions. What is the end-to-end performance of ZaliQL for causal inference in observational data? Does ZaliQL support advanced methods for causal inference and produce results with the same quality as statistical software? How does ZaliQL scale up with increasingly large data sets? And how effective are the optimization techniques in ZaliQL?

## 5.1 Setup

**Data** The *flight dataset* we used was collected by the US Department of Transportation (U.S. DOT) [**?**]. It contains records of more than 90% of US domestic flights of major airlines from 1988 to the present. Table **??**(shown in Section 1) lists dataset attributes that are relevant to our experiments. We restrict our analysis to about 105M data entry collected between 2000 and 2015.

The *weather dataset* was gathered using the weather underground API [**?**]. Its attributes are also presented in Table **??**. In addition, we pre-computed two other attributes AiportTraffic and CarrierTraffic. The former is the total number of flights occur in the origin airport of a flight one hour prior to the flight departure time, the latter is the same quantity restricted to the flights from the same carrier. We managed to acquire and clean 35M weather observations between 2000 and 2015. These datasets are integrated by a spatio-temporal join.

**Causal questions and covariate selection** We explore the causal effect of the following binary treatments on flight departure delays and cancellation: LowVisibility (1 if Visim< 1; 0 if Visim> 5); Snow (1 iff Precipm> 0.3 and Tempm< 0); WindSpeed (1 if Wspdm> 40; 0 if Wspdm< 20); and Thunder. In each case, data items where the attribute in question was in between the two bounds were discarded. For instance, for the treatment LowVisibility, we want to assess the following counterfactual: *"What would the flight departure delay have been, if visibility were fine, i.e., Visim> 5, when visibility is actually low, i.e., Visim< 1"*; for this analysis, items where Visim$\in [1, 5]$ were discarded.

To ensure the SUTVA (cf. Section 2), we considered the difference between the actual delay and the late aircraft delay (if one exists) as the outcome of interest. Therefore, we assumed that the potential delay of each flight did not depend on the treatment assignment to the other flights namely, there was no interference between the units.

To obtain quality answers for each treatment, we used graphical models to identify a minimum number of covariate attributes to ensure unconfoundedness, because minimizing the number of covariates has been shown to increase the precision of the matching estimators [**?**]. We used the tool provided by `http://dagitty.net` to construct the causal DAG (CDAG) and select the minimum number of covariates that separate the treatment assignments from the potential outcomes. The tool finds a minimal subset of variables $X$ that forms a *d-separation* [**?**] of the treatment $T$ from the effect $Y$ (meaning: all paths from $T$ to $Y$ go through some variable in $X$). For example, Figure 7 shows the CDAG for the treatment LowVisibility and the effect DepDelay: the

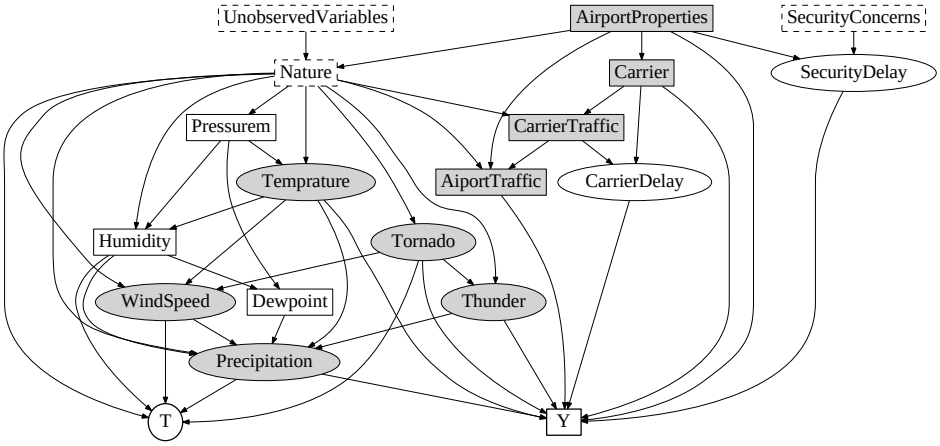
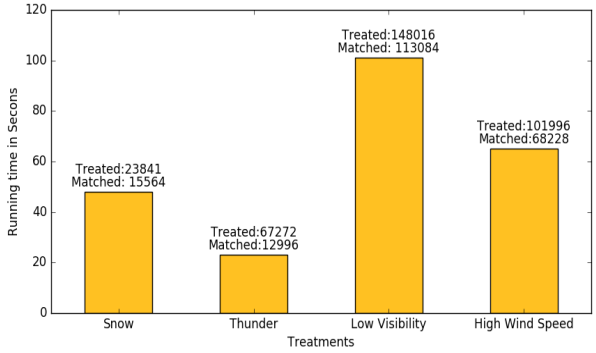
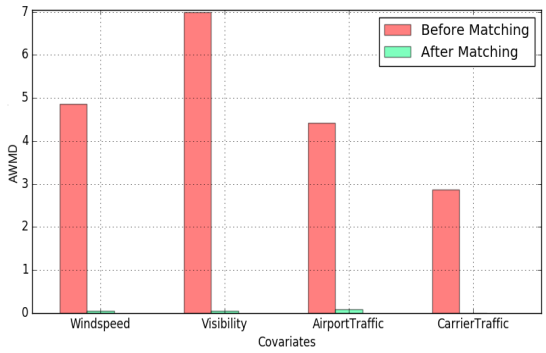


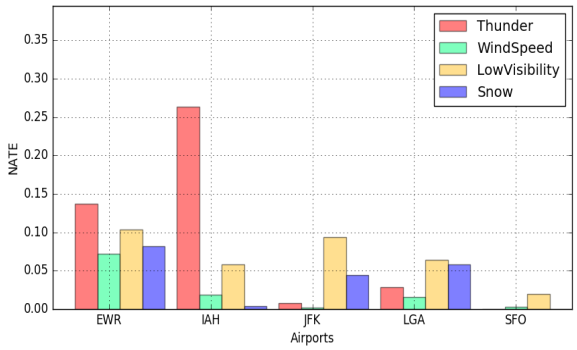

Figure 7: **A Causal Directed Acyclic Graph (CDAG) demonstrating the assumptions necessary to estimate $\tau_{ATE}$ of the treatment T(LowVisibility) to the outcome Y(DepDelay).** A CDAG shows how conditioning on observable covariates (X) breaks the confounding causal relationships (e.g., T ← Precipitation → Y) and allows the estimation of $\tau_{ATE}$. In other words, we assume that the treatment assignment $T$ and potential outcomes $Y = Y(z)$ for z=0,1 are unconfounded by conditioning on the filled nodes which is a minimum cardinality set of *observed* variables that $d$-sperate the two. Attributes in boxes are from the flight data; Those in ovals are from the weather data; Those in dashed boxed are unobserved.



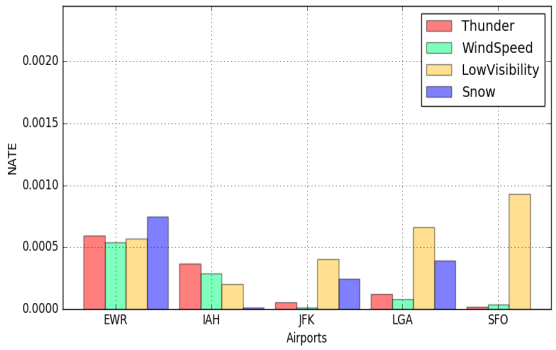

a) Running Time of Matching.




b) Measuring Imbalance Reduction.

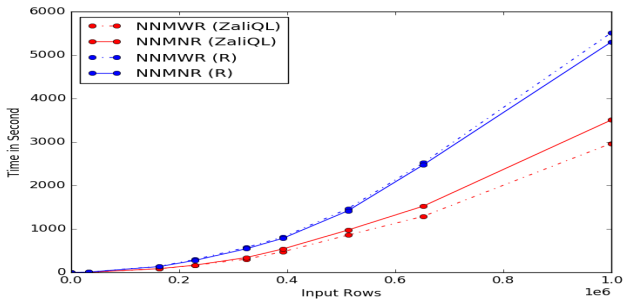



c) Causal Effect of Weather Types on Departure Delay.

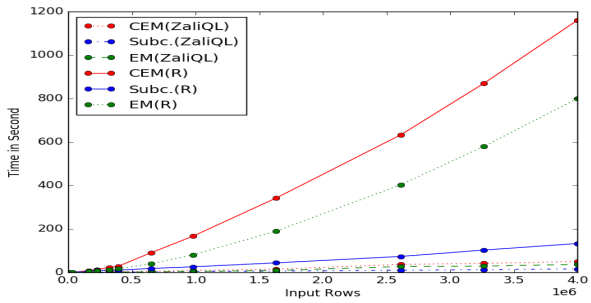



d) Causal Effect of Weather Types on Cancellation.

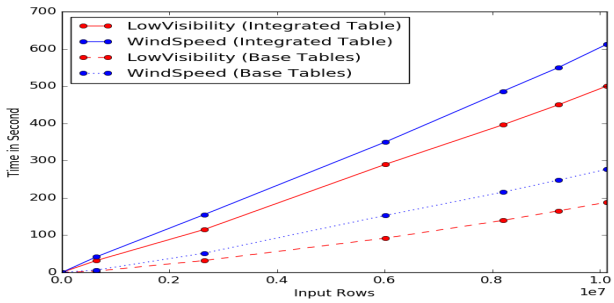
Figure 8: **Analysis of the causal effect of different weather types on flight departure delay and cancellation.**



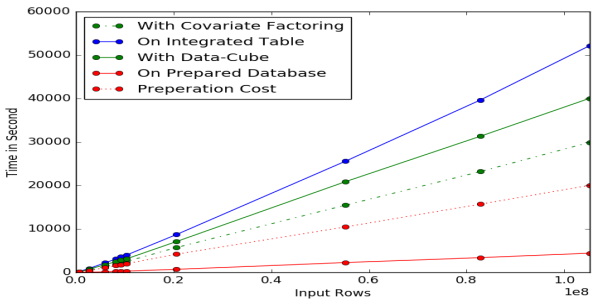

a) Scalability Comparison (NNMWR and NNMNR).




b) Scalability Comparison (CEM, EM and Subclas.).




c) Efficacy of Pushing Matching.




d) Efficacy of optimizations for multiple treatments.

Figure 9: **Scalability and optimizations Evaluation.**

| | | | | Absolute Weighted Mean Difference (AWMD) | | | |
|---|---|---|---|---|---|---|---|
| **Method** | | **Control** | **Treated** | **Visibility** | **WindSpeed** | **AirportTraffic** | **CarrierTraffic** |
| | *Raw Data* | 214457 | 464 | 12.7529 | 3.6363 | 4.5394 | 2.9465 |
| **NNMWR** | R | 311 | 323 | 0.0724 | 0.8789 | 0.6935 | 0.2724 |
| | ZaliQL | 296 | 312 | 0.0692 | 0.5756 | 0.6955 | 0.8044 |
| **NNMNR** | R | 318 | 318 | 0.1006 | 1.0308 | 0.3396 | 0.1352 |
| | ZaliQL | 291 | 291 | 0.0769 | 0.7216 | 0.3195 | 0.5910 |
| **Subclass.** | R | 1275 | 255 | 2.5054 | 1.4631 | 1.6013 | 1.0022 |
| | ZaliQL | 1002 | 255 | 0.0684 | 1.0631 | 0.1872 | 0.0905 |
| **EM** | R | 8 | 7 | 0 | 0 | 0 | 0 |
| | ZaliQL | 8 | 7 | 0 | 0 | 0 | 0 |
| **CEM** | R | 2284 | 340 | 0.2875 | 0.0542 | 0.1135 | 0.0905 |
| | ZaliQL | 2284 | 340 | 0.28755 | 0.0542 | 0.1135 | 0.0905 |

Table 2: **Quality Comparison between ZaliQL and R.**

set $X$ consists of the shaded nodes d-sperate the treatment assignment and the potential outcomes.

**Systems** The experiments were performed locally on a 64-bit OS X machine with Intel Corei7 processor (16 GB RAM, 2.8 GHz). ZaliQL was deployed to Postgres version 9.5. We compared ZaliQL with R packages MatchIt and CEM, version 2.4 and 1.1 respectively, available from [**?**].

## 5.2 Results

### 5.2.1 *End-to-End Performance*

We estimated the causal effect of different weather types on flight departure delay and cancellation at five major US airports, that are among the ten airports with the worse weather-related delays according to [**?**], namely: San Francisco (SFO) , John F. Kennedy (JFK), Newark Liberty (EWR), George Bush (IAH), and LaGuardia Airport (LGA). The flight and weather data associated with these airports consists of about 10M and 2M rows, respectively.

To this end, we estimated $\tau_{ATE}$ for each treatment associated to a weather type by performing CEM wrt. its identified covariates. Continuous covariates were coarsened into equal width buckets, and categorical covariates were matched by their exact values.

Figure 8(a) reports the running time of performing CEM for each treatment: recall (Fig. 5) that this involves a group-by followed by an elimination of all groups that have no treated, or no control unit. Next, we evaluated the quality of the CEM, in Figure 8(b), using a standard metric in the literature: the *absolute weighted mean difference (AWMD)* of each continuous covariate between the treated and control group:

$$\mathbb{E}_b[|\mathbb{E}[x_i|T=1, B(X)=b] - \mathbb{E}[x_i|T=0, B(X)=b]|] \qquad (5)$$

for each covariate attribute $x_i \in X$. This is a standard measure of imbalance, since in a perfectly balanced group the difference is 0 (see Eq. 3). Note that in the case of CEM, we assume subclasses form the balancing scores. We compared this difference for the original value of the covarites before and after CEM. As shown, CEM substantially reduced the covariates imbalance in the raw data: this graph shows the perils of attempting to do causal inference naively, without

performing CEM or matching. We also observed that CEM results in quite reasonable matched sample size wrt. treatments ( more than 75% of the treated units are matched with the average rate of one treated to five control units).

Next, Figure 8(c) shows the causal effect of weather type on departure delay: it shows the estimated $\tau_{ATE}$ for each target airport, normalized by the number of treated units to the total number of units in each aiport. Following common practice in causal inference we estimated $\tau_{ATE}$ using Eq. 4 (thus assuming that CEM produces perfectly balanced groups). The normalization lets us compare the effects of different weather types in an individual airport. In fact, $\tau_{ATE}$ reflects the potential causal effect of a treatment to an outcome. For instance, IAH barely experiences snow, but snow has a devastating effect on flight delay at this airport. The results in Figure 8(c) are in line with those in [**?**], which reported the following major weather-related causes of flight delay at the airports under the study: snowstorms,thunderstorm and wind at EWR; thunderstorm and fog at IAH; snowstorms and visibility at JFK; snowstorms at LGA; fog and low clouds at SFO;

Figure 8(d) a similar causal effect of weather type on a different outcome, namely Cancellation. Here, we leveraged the fact that matching and subclassification do not depend on a particular outcome variable, thus, a matched sample wrt. a treatment can be used to estimate its causal effect of several outcomes.

#### 5.2.2 Quality Comparisons with R

The MatchIt and CEM packages in R are popular tools for conducting causal inference today. In this section we compare the quality of the results returned by ZaliQL with those returned by the R packages. We considered all types of matchings described in Sec. 3: NN matching (both NNMNR and NNMWR), and subclassification (by propensity score, CEM and exact matching(EM)). Sine the R packages do not scale to large datasets, we conducted these experiments over a random sample of data used in Section 5.2.1, which consists of around 210k rows. We evaluated different matching methods for the treatment of Snow.

Table 2 compares the size of the matched sample and the AWMD (Eq.5) obtained by ZaliQL and R, for different matching methods. These results shows that ZaliQL produced results whose quality was at least as good as R. Note that for NNMNR and NNMWR, the caliper 0.001 is applied. For subclassification all units with propensity score less than 0.1 and more than 0.9 are discarded (this is a common practice in causal inference). We observe that all matching methods produce well-balance matched samples with reasonable sizes. However, among these methods CEM works better both in terms of the imbalance reduction and size of the matched sample.

The slight differences between the results of NNM arose from a slight difference between the propensity score distribution we obtained using logistic regression provided by MADlib inside Postgres.

#### 5.2.3 Scalability Testing

We compared the scalability of different matching methods in ZaliQL and R. The experiment was carried out over a random samples of data used in Section 5.2.1 and for the treatment of Snow. Figure 9(a) compares NNM methods based on propensity score. We observe that NNM does not scale to large data. Note that, for the reasons mentioned in Section 3.1, optimizing this method was not the aim of this paper. Figure 9(b) compares CEM, EM and subclassification in ZaliQL and R. Note that for CEM and NNMWR in ZaliQL we respectively implemented the group-by and window function statement. As depicted, ZaliQL significantly outperforms R and scale to large data.

#### 5.2.4 Efficacy of the Optimization Techniques

We tested the effectiveness of the proposed optimization techniques (cf. Section 4). Figure 9(c) compares the running time of performing CEM on the integrated weather and flight tables with CEM on base tables (cf. Section 4.2) for two treatment LowVisibility and WindSpeed. The analysis was carried out on data used in Section 5.2.1. Note that the cost of integrating two tables is ignored.

For covariates factoring and data-cube optimizations, we compared the total cost of performing matching on the treatments defined in Section 5.1, plus the treatment obtained by conjunction of Snow and WindSpeed, which we refer to as Snowstorm. This analysis was carried on the entire integrated dataset. By applying Algorithm 3, the treatments are partitioned into two groups, namely $g_0$={Snow, WindSpeed, Snowstorm} and $g_2$={LowVisibility, Thunder}. Figure 9(d) compares the total running time of matching for each treatment after covariate factoring with the naive matching procedure. Figure 9(d) also shows the total cost of matching with and without using data-cubes. In addition, it represents the total cost of matching on the prepared database, according to Algorithm 4, and the cost of database preparation. As depicted, matching on the prepared database reduce the over cost of matching by an order of magnitude.

## 6. RELATED WORK AND CONCLUSION

The simple nature of the R, and its adherence to a few statistical assumptions, makes it more appealing for the researchers. Therefore, it has become the prominent approach in social sciences, biostatistics, political science, economics and other disciplines. Many toolkits have been developed for performing casual inference *a' la* this framework that depends on statistical software such as SAS, SPSS, or R project. However, these toolkits do not scale to large datasets. This work introduce ZaliQL, a SQL-based framework for drawing causal inference that circumvents the scalability issue with the existing tools. ZaliQL supports state-of-the-art methods for causal inference and runs at scale within a database engine.

The notion of causality has been studied extensively in databases [**?**, **?**, **?**, **?**]. We note that this line of work is different than the present paper in the sense that, it aims to identify causes for an observed output of a data transformation. While these works share some aspects of the notion of causality as studied in this paper, the problems that they address are fundamentally different.