Probabilistic 'Best Set' Sorting Algorithms for Multivariate (Prognostic) Data.

Fang Cao, Kobi Abayomi, Nagi Gebraeel

***Abstract*—We derive a general method for the probabilistic identification of the best (worst) k of n multivariate (m dimensional) cases. As a statistics problem, i.e. to generate a predicted best (worst) set, we recast two deterministic sorting algorithms — the Threshold and Minimal Probing algorithms — as probabilistic queries, subject to tunable tolerance levels, that converge to the deterministic best (worst) sets. Our investigation is motivated by identification of the 'top (bottom) - k' units in a multiple machine, multiple component system that generates prognostic failure information. By exploiting distributional and dependency assumptions among the components our probabilistic algorithm yields a less 'costly' probe of the machines. More generally: we offer probabilistic versions of well known sorting algorithms and demonstrate bounds for probabilistic multivariate sorting based on between case dependence, via the copula and multivariate order statistics.**

I. Introduction

The setting for this paper is the identification of the best (or worst) k units in a fleet of n m-component machines; this is referred to as a 'top-k' search [(5)]. The goal is to find the top-k units using the least amount of information: the fewest machines and minimal number of components in each machine. More generally, this is to sort multivariate data while inspecting as little of it as possible; this is exactly the task in the well-known threshold algorithm (TA) [(6)] and its revision, the minimal probing (MPro) [(12)] algorithm. These algorithms are prototypically deterministic: the sorting proceeds without any distributional assumptions on the data and only the most general concatenation of the component information [(20), (21), (1)]. Our innovation is to recast these algorithms under distributional assumptions, and under more general functions for 'scoring' each machine (case, unit).

We specifically consider the situation where the units (machines, cases) offer degradation or health information from which we can generate 'prognostics': residual lifetime predictions [(13), (24)]. Prognostics deals with predicting the remaining useful life of partially degraded systems. In a few applications, direct measurement of the degradation state of a component or system is possible, in most cases however such measurements are not accessible. For these cases, indirect measurements of the degradation states are often the primary, if not the only, way to assess the state of health of a system during its operation [(25)].

These indirect measurements are the multivariate data on which we perform the 'top-k' sorting; the data are 'degradation signals' — patterns related with physical transitions in degradation. These signals are used to estimate the remaining, or residual life, of the system components [(9), (18)]. Acquisition of these data is often costly: the desirable goal in a 'top-k' procedure is to find the best (worst) units (machines, cases) with the minimal cost from data collection. Gebraeel considers the standard, deterministic, TA and MPro algorithms in (8): this paper is a generalization of that approach under proper, probabilistic assumptions, and in particular a specification of the degradation data as random observations.

The methodology in this paper is not limited to the degradation setting: the algorithms we offer can readily and immediately be applied to any setting for probabilistic sorting of multivariate, dependent, data.

All Authors: *Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA*. Corresponding Author, email: *kobi@gatech.edu*

A. Commensurate work

Recent work [(4)] describes an approach to compute a top-k set of answers (analogous to our cases, units, machines) to an SQL query on a probabilistic database: this set is then sorted by probability. This algorithm runs several Monte-Carlo simulations in parallel, one for each candidate answer, and approximates each probability only to the extent needed to compute correctly the top-k answers. Soliman also [(14)] studies the top-k query in an 'uncertain' database. This paper integrates traditional top-k semantics with "possible worlds" semantics, and constructs a framework that encapsulates a probabilistic model and efficient query processing techniques. While the paper demonstrates a minimal number of accessed m-tuples the algorithms has exponential complexity in both time and component-wise space. In affix to (14), (11) introduces polynomial algorithms for processing top-k queries in uncertain databases under the generally adopted model of x-relations, and the algorithms run in near linear or low polynomial time in uncertain databases.

In (2) and (16) model-driven approaches peculiar to sensor networks are used in conjunction with top-k type query processing. Our technique differs from these in that it is explicitly probabilistic; we rely on multivariate distributional assumptions for the data, given the (particular degradation) model we outline below.

Lastly, and in direct appeal to the TA/MPro algorithms, (15) and (23) have introduced approximate versions via 'guaranteed' thresholding of the error of selecting a non-top-k unit. Our approach differs because we explicitly score each unit (case, machine), offer general results for the choice of component-wise concatenation/scoring, and yield almost sure convergence to the deterministic result under distributional assumptions.

This paper consists of six additional sections. The second section, below, discusses the particular degradation model we suppose to generate the data. The third and fourth sections introduce the random variable based algorithm for top-k identification, including probing strategy and stopping criterion, for the TA and MPro algorithms. We illustrate some results from (simulated) prognostic data in the fifth section, comment more generally on multivariate probabilistic sorting under between case dependency in the sixth section and consider further work in the conclusion. Figures follow the body of the paper. The methods we describe in the third, fourth and sixth sections are immediately generalizable to any setting where sorting/ranking of multivariate-dependent data is desirable; the second section can be considered modular: peculiar to the specific data we consider here and readily exchangeable.

II. Degradation Modeling

The typical assumptions about the degradation process, and thus the data, are 'graceful' degradation and monitoring of the deterioration via dedicated sensors. If the units (machines, cases) exhibit a predictable degradation pattern, future health can be predicted from the collected information. Since direct measurements of the degradation states are often not accessible, indirect measurements are often the primary if not the only way to assess the state of health of a system during its operation. This is performed using sensor-based condition monitoring techniques in which degradation or performance measures, such as vibration, temperature, and acoustic emissions, are observed over time.

These observed measures typically exhibit characteristic patterns associated with the underlying physical transitions occurring during degradation. The measurements are known as degradation signals. Given a predetermined failure threshold, degradation signals can be used to estimate the residual life of partially degraded components. Therefore the degradation model combines two sources of information: the reliability characteristics of the population of critical components and real-time sensor information.

A. Degradation model

A common type of degradation model is the exponential model with Brownian motion error:

$$L(t) = \theta' + \beta' t + \epsilon(t) \tag{II.1}$$

where $L(t)$ is the logged degradation signal, θ' is the random intercept with prior distribution

$N(\mu_0, \sigma_0^2)$, and β' is the random slope with prior distribution $N(\mu_1', \sigma_1^2)$ with $\mu_1' = \mu_1 - \frac{\sigma^2 t}{2}$ and $\epsilon(t)$ following a discrete time Wiener process with mean zero and variance $\sigma^2 t$.

At time 'epochs' $t_1, t_2, ..., t_k$ where

$$t_j = j \cdot t_1, j = 1, ..., k$$

the corresponding signals are $L_1, ..., L_k$.

Given the observed signals $L_1, ..., L_k$, the posterior distribution of β' is $N(\mu_{\beta'}, \sigma_{\beta'}^2)$, and the posterior distribution of θ' is $N(\mu_{\theta'}, \sigma_{\theta'}^2)$, where

$$\mu_{\theta'} = A^{-1}[(L_1\sigma_0^2 + \mu_0\sigma^2)(\sigma_1^2 t_k + \sigma^2) \quad \text{(II.2)}$$

$$-\sigma_0^2 t_1(\sigma_1^2 \sum_{i=1}^{k} L_i + \mu_1'\sigma^2)] \quad \text{(II.3)}$$

$$\mu_{\beta'} = A^{-1}[(\sigma_1^2 \sum_{i=1}^{k} L_i + \mu_1'\sigma^2)(\sigma^2 t_1 + \sigma_0^2) \quad \text{(II.4)}$$

$$-\sigma_1^2(\sigma_0^2 L_1 + \mu_0\sigma^2 t_1)] \quad \text{(II.5)}$$

$$\sigma_{\theta'}^2 = A^{-1}[\sigma^2\sigma_0^2 t_1(\sigma_1^2 t_k + \sigma^2)] \quad \text{(II.6)}$$

$$\sigma_{\beta'}^2 = A^{-1}[\sigma^2\sigma_1^2(\sigma_0^2 + \sigma^2 t_1)] \quad \text{(II.7)}$$

with

$$A = (\sigma_0^2 + \sigma^2 t_1)(\sigma^2 + \sigma_1^2 t_k) - \sigma_0^2\sigma_1^2 t_1).$$

For the above degradation model, the parameters $\mu_0, \mu_1, \sigma_0, \sigma_1, \sigma$ can be estimated using observed signals [(9)]. This can be seen as a a particular version of a hierarchical linear or 'random-effects' model [(10)]: the model is fixed but the effects/coefficients are random and updated from observable data.

B. 'Health' measurement:

According to the model (II.1), we have the following equations:

$$L(t_k) = \theta' + \beta' t_k + \epsilon(t_k) \quad \text{(II.8)}$$

$$L(0) = \theta' + \epsilon(0) \quad \text{(II.9)}$$

where is t_k the kth time epoch. Equations (II.8) and (II.9) yield

$$\tilde{\beta}' \triangleq \frac{L(t_k) - L(0)}{t_k} = \beta' + \frac{\epsilon(t_k) - \epsilon(0)}{t_k}, \quad \text{(II.10)}$$

where $\beta' \sim N(\mu_{\beta'}, \sigma_{\beta'}^2)$; $\frac{\epsilon(t_k)-\epsilon(0)}{t_k} \sim N(0, \frac{\sigma^2}{t_k})$. Thus $\tilde{\beta}' \sim N(\mu_{\beta'}, \sigma_{\beta'}^2 + \frac{\sigma^2}{t_k})$ due to independence.

Based on the above deduction, we define the 'health' as

$$X \triangleq \frac{\tilde{\beta}'}{\mu_{L(t_k)} - D} \quad \text{(II.11)}$$

by algebra and the normality of linear transforms of normal random variables

$$X \sim N\left(\frac{\mu_{\beta'}}{\mu_{\theta'} + \mu_{\beta'} t_k - D}, \frac{\sigma_{\beta'}^2 + \frac{\sigma^2}{t_k}}{(\mu_{\theta'} + \mu_{\beta'} t_k - D)^2}\right), \quad \text{(II.12)}$$

where D is the predetermined threshold value, and $\mu_{L(t_k)} = \mathbb{E}[L(t_k)] = \mu_{\theta'} + \mu_{\beta'} t_k$.

In a physical interpretation of the formula, the score X is an approximation for the reciprocal of the negative residual life time; the larger score, the 'healthier' the component.

Once the 'health' scores are calculated, the can be used as inputs to a sorting algorithm to identify the top-k units. We introduce a probabilistic version of the TA algorithm, in the next section, and then of the MPro algorithm. Both versions rest upon an elucidation of the data — the health scores — as proper observations: that is as instantiations of random variables.

III. Probabilistic top-k query based on the Threshold Algorithm

The TA algorithm for top-k ranking was first proposed by Fagin in 1996 [(6)]. The algorithm begins under the assumption that each column of the data is ordered. The algorithm proceeds by scanning the sorted data row-by-row, i.e. case by case (unit, machine), and stops when the threshold value of the

row is no greater than the minimal overall score of current top-k list. Nevertheless, TA treats the scores as deterministic values, and does not incorporate the distributional information of the scores. Here we develop a probabilistic algorithm by leveraging the dependence among components. This yields a probabilistic TA procedure which converges to the deterministic TA result, but requires less iterations at less than certain levels of tolerance. We recommend either (6) or (8) for a full take on the TA algorithm.

A. The data(base) as random variable(s): $\mathbf{X}$

Suppose we have n units, each with m components. Let real valued $\mathbf{X}$ be the entire data; X_{ij} the 'health' of the j^{th} component of i^{th} unit. For each unit (case, machine), say, real vector valued $\mathbf{X}_i$ can follow multivariate normal distribution; this is to assume the units (cases, machines) are independent from one another (across n) but retain inter-component dependence (across m). It should be noted that we could consider any elliptical multivariate distribution (multivariate exponential distribution (MVE) or multivariate inverse Gaussian distribution (MIG), for example), i.e. any distribution with second order dependence.

Let $X_{(k)j}$ be the k^{th} order statistic for column j such that $X_{(n)j} = \max_{1\le i\le n} X_{ij}$ and $X_{(1)j} = \min_{1\le i\le n} X_{ij}$. The goal to derive the distribution for the random vector $\mathbf{X}_{(r)} = [X_{(r)1}X_{(r)2}, ..., X_{(r)m}]$, as well as certain functions of it. This is the crux of the probabilistic extension of the TA algorithm: the derivation of the distribution of the multivariate order statistics, the rth of n independent, sorted, rows. We use this result to predict the stopping 'place' (iteration) for the Threshold Algorithm, and use that as the objective.

B. Deriving $F_{X_{(r)}}$, the $c.d.f.$ for $X_{(r)}$

For illustration we start with the two-component case. The derivation mimics the problem of putting n balls into 4 bins with constraints, as illustrated in Table III-B.

X_{i1}	X_{i2}		Total
	$\le x_2$	$> x_2$	
$\le x_1$	t_1	t_2	l_1
$> x_1$	t_3	t_4	$n-l_1$
Total	l_2	$n-l_2$	n

Table I
DERIVING THE DISTRIBUTION OF MULTIVARIATE ORDER STATISTICS WHEN $m = 2$

The derivation:

$$F_{X_{(r)}}(x_1,x_2) = \mathbb{P}(X_{(r)1}\le x_1, X_{(r)2}\le x_2)$$

$$= \mathbb{P}(\text{At least } r \text{ of } X_{i1}\le x_1, \text{at least } r \text{ of } X_{i2}\le x_2)$$

$$= \sum_{l_1,l_2\ge r} \mathbb{P}(\text{Exactly } l_1 \text{ of } X_{i1}\le x_1, \text{exactly } l_2 \text{ of } X_{i2}\le x_2)$$

$$= \sum_{\substack{0\le t_1,t_2,t_3,t_4\le n\\ t_1+t_2+t_3+t_4=n\\ r\le l_1,l_2\le n}} \frac{n!}{t_1!t_2!t_3!t_4!} p_1^{t_1}p_2^{t_2}p_3^{t_3}p_4^{t_4} \quad \text{(III.1)}$$

where

$$p_1 = \mathbb{P}(X_{i1}\le x_1, X_{i2}\le x_2), \quad \text{(III.2)}$$
$$p_2 = \mathbb{P}(X_{i1}\le x_1, X_{i2} > x_2), \quad \text{(III.3)}$$
$$p_3 = \mathbb{P}(X_{i1} > x_1, X_{i2}\le x_2), \quad \text{(III.4)}$$
$$p_4 = \mathbb{P}(X_{i1} > x_1, X_{i2} > x_2). \quad \text{(III.5)}$$

So, for the special cases of the above result, the $c.d.f.$ of the "top row" is

$$F_{X_{(n)}}(x_1,x_2) = p_1^n \quad \text{(III.6)}$$

and the $c.d.f.$ of the "bottom row" is

$$F_{X_{(1)}}(x_1,x_2) = 1-(p_2+p_4)^n-(p_3+p_4)^n+p_4^n. \quad \text{(III.7)}$$

This can immediately be generalized to m-component case following the above outline. Let $c = 2^m$, and the result is as follows

$$F_{X_{(r)}}(x_1,x_2,...,x_m)$$
$$=\mathbb{P}(X_{(r)1}\le x_1, X_{(r)2}\le x_2, ..., X_{(r)m}\le x_m)$$

$$= \sum_{\substack{0\le t_1,t_2,...,t_c\le n \\ t_1+t_2+..+t_c=n \\ r\le l_1,l_2,...,l_m\le n}} \frac{n!}{t_1!t_2!...t_c!} p_1^{t_1} p_2^{t_2} ... p_c^{t_c}, \quad \text{(III.8)}$$

(III.9)

where $p_1, p_2, ..., p_c, l_1, ..., l_m$ are defined in the same manner as two-component case.

C. Deriving the distribution of functions of $X_{(r)}$

After obtaining the $c.d.f.$ of $\mathbf{X}_{(r)}$, we need to derive the distribution of scoring functions $\tau(.)$ of $\mathbf{X}_{(r)}$. Typically, the 'scoring' function $\tau(.)$ is a monotone function like max, min, sum [see (8)]. Let G_r denote the $c.d.f.$ of $\tau(\mathbf{X}_{(r)})$. When τ is the max function,

$$\begin{aligned} G_r(s) &= \mathbb{P}(\tau(\mathbf{X}_{(r)}) \le s) && \text{(III.10)} \\ &= \mathbb{P}(\max_{1\le j\le m} X_{(r)j} \le s) \\ &= \mathbb{P}(\cap_{j=1}^m X_{(r)j} \le s) \\ &= F_{X_{(r)}}(s, s, ..., s). && \text{(III.11)} \end{aligned}$$

Similarly, when τ is the min function,

$$\begin{aligned} G_r(s) &= \mathbb{P}(\tau(\mathbf{X}_{(r)}) \le s) && \text{(III.12)} \\ &= \mathbb{P}(\min_{1\le j\le m} X_{(r)j} \le s) \\ &= \mathbb{P}(\cup_{j=1}^m X_{(r)j} \le s) \\ &= 1 - \mathbb{P}(\cap_{j=1}^m X_{(r)j} > s) \\ &= 1 - \bar{F}_{X_{(r)}}(s, s, ..., s), && \text{(III.13)} \end{aligned}$$

where $\bar{F}_{\mathbf{X}_{(r)}}$ is the survival function of $\mathbf{X}_{(r)}$. For the distribution of the sum, for example, of the r^{th} row, we just adopt the linear transformation on the $X_{(r)}$, then take limit with respect to other variables. This is just the ordinary change of variable method which enables us to get the desired $c.d.f.$ for monotone functions $\tau(.)$ of each rth row.

From the distribution of $\tau(\mathbf{X}_{(r)})$, we can compute the probability that we have found the "true" top-k units at each row. Suppose t_r is the minimum overall score of the current top-k list before probing the r^th row, then

$$\begin{aligned} p_r &\triangleq \mathbb{P}(\text{The algorithm stops at } r^{th} \text{ row}) \\ &= \mathbb{P}(\tau(\mathbf{X}_{(r)}) \le t_r) = G_r(t_r). \quad \text{(III.14)} \end{aligned}$$

Theorem III.1. Assume $\tau : \mathbb{R}^m \to \mathbb{R}$ is a non-decreasing function. Then p_r is nondecreasing with respect to r.

Proof: When $r \in 1, 2, ..., n-1$ then $r+1 \in 2, ..., n$. As we mine the data 'deeper', we have $t_{r+1} \ge t_r$ since the top-k list will be updated only when the overall score of a 'new' machine is larger than that of one of the existing machines. As they are order(ed) statistics, $X_{(r)j}$ is larger than $X_{(r+1)j}$ in stochastic order, $\forall j = 1, ..., m$. Since τ is non-decreasing function, then $\tau(X_{(r)})$ is larger than $\tau(X_{(r+1)})$ in stochastic order [see (17)].

From the definition of stochastic order,

$$\begin{aligned} p_r &= \mathbb{P}(\tau(\mathbf{X}_{(r)}) \le t_r) && \text{(III.15)} \\ &\le \mathbb{P}(\tau(\mathbf{X}_{(r)}) \le t_{r+1}) \\ &\le \mathbb{P}(\tau(\mathbf{X}_{(r+1)}) \le t_{r+1}) = p_{r+1}. \end{aligned}$$

Hence the result holds. ■

Using the monotone property of p_r, with a predetermined threshold confidence level α, we can stop the the algorithm when $p_r \ge \alpha$. In this way, we can find the top-k units with desired confidence by scanning fewer rows than the TA.

D. Probabilistic TA algorithm

The algorithm then is a probabilistic augmentation of the top-k ranking algorithm based on TA.

Algorithm

1) Rank each attribute of the original database in descending order
2) Probe the "sorted" database row-by-row from the top, and find the current top-k units after probing each row
3) Compute the minimum of the overall scores of the current top-k units
4) Repeat step 2 and 3 until the probability of finding the top-k exceeds the threshold, i.e., $\mathbb{P}(g(X_{(r)}) \le t_r) \ge \alpha$.

Table II
PROBABILISTIC THRESHOLD ALGORITHM

IV. Probabilistic top-k query based on MPro algorithm

The Minimal Probing (MPro) algorithm [(12)] was developed to minimize probe 'cost' for top-k queries. MPro differs from TA chiefly in the relaxation of the sorting prerequisite: in MPro the data are arrive unsorted and the goal is to 'order' as minimally as possible. The MPro algorithm, then, is a recipe for returning the top-k units (cases, machines) with iterative sorting of particular data. In the deterministic case, MPro is demonstrated to be probe-optimal: i.e. that the prescribed order for inspecting units has the minimum iterations. Nevertheless, the algorithm is designed for deterministic values and ignores any distributional information for the data. We introduce here a probabilistic version of the MPro algorithm which minimizes the probing 'cost' for the top-k query by explicitly assuming the data are random variables. We illustrate our augmented algorithm (probing strategy) with an associated objective (stopping criterion), based on an explicit assumption about the joint distribution of the data.

A. Explicitly modeling the data

As before, suppose X_{ij} is the score of the j^{th} component of the i^{th} machine, $X_{ij} \sim N(\mu_{ij}, \sigma_{ij}^2)$, $i = 1, ..., n, j = 1, ..., m$. Here we need to also make explicit the correlation structure between the components, that is, $Corr(X_{ij}, X_{ij*}) = Corr(X_{kj}, X_{kj^*})$ for all i, j, j^*, k.

Thus the distribution of scores of the i^{th} machine is $\mathbf{X}_i \sim N(\mu_i, \Sigma_i)$, where

$$\begin{aligned}
&\mu_i = (\mu_{i1}, \mu_{i2}, ..., \mu_{im}),\\
&\Sigma_i = V_i^{1/2} R V_i^{1/2},\\
&V_i = diag(\sigma_{i1}^2, \sigma_{i2}^2, ..., \sigma_{im}^2),\\
&R_{jk} = Corr(X_{ij}, X_{ik}) \triangleq \rho_{jk}.
\end{aligned}$$

Let $X_{ij}^* = \frac{X_{ij} - \mu_{ij}}{\sigma_{ij}}$, then $X_{ij}^* \sim N(0, 1)$. If we denote $\mathbf{X}_i^* = (X_{i1}^*, X_{i2}^*, ..., X_{im}^*)$, $\mathbf{X}_i^*$'s will be i.i.d multivariate normal, and the correlation matrix of $\mathbf{X}_i^* = (X_{i1}^*, X_{i2}^*, ..., X_{im}^*)$ is

$$R = \begin{bmatrix} 1 & \cdots & \rho_{1m} \\ \vdots & \ddots & \vdots \\ \rho_{m1} & \cdots & 1 \end{bmatrix}.$$

We can use sample correlation matrix of the standardized variables as our estimate for R. In practice, R is estimated from historical (training) data and could be used for validation.

The essential difference between the MPro and TA algorithms is that the entire row of ordered data is not available in MPro as we iterate. Thus, we need to calculate some sort of conditional information at each iteration and let the units (cases, machines) enter the 'top-k' set based on this conditional criterion.

B. Probing strategy

For TA, we have to probe the data row-by-row; all components in a row (machine, unit, case) are inspected simultaneously. In MPro we can probe the components more flexibly.

Suppose the scoring function is the sum, i.e., the overall score of the i^{th} machine is

$$\tau_i = \sum_{j=1}^{m} X_{ij}. \tag{IV.1}$$

Assume the order of the components to probe is the same for all inspected machines (units, cases) at each iteation. After probing r components, the conditional expectation is

$$\mathbb{E}(\tau_i | \mathbf{X}_{ir}) = \sum_{j=1}^{r} X_{ij} + \sum_{j=r+1}^{m} \mathbb{E}(X_{ij} | \mathbf{X}_{ir}). \tag{IV.2}$$

Since X_{ij} and $\mathbf{X}_{ir}$ are jointly normally distributed,

$$\mathbb{E}(X_{ij} | \mathbf{X}_{ir}) = \mu_{ij} + \Sigma_{ijr} \Sigma_{irr}^{-1} (\mathbf{X}_{ir} - \mu_{ir}),$$

where

$$\mathbf{X}_{ir} = (X_{i1}, X_{i2}, ..., X_{ir}),$$

$$\mu_{ir} = (\mu_{i1}, \mu_{i2}, ..., X\mu_{ir}),$$
$$\Sigma_{ijr} = Cov(X_{ij}, \mathbf{X}_{ir}),$$
$$\Sigma_{irr} = Var(\mathbf{X}_{ir}).$$

The conditional variance is

$$\begin{aligned}\mathrm{Var}(g_i|\mathbf{X}_{ir}) &= \mathrm{Var}(\sum_{j=r+1}^{m} X_{ij}|\mathbf{X}_{ir}) \\ &= Var(Y_i|\mathbf{X}_{ir}) \\ &= \Sigma_{iyy} - \Sigma_{iyr}\Sigma_{irr}^{-1}\Sigma'_{iyr},\end{aligned}$$

where

$$Y_i = \sum_{j=r+1}^{m} X_{ij},$$
$$\Sigma_{iyy} = \mathrm{Var}(Y_i) = \sum_{1\le j,k\le m} Cov(X_{ij}, X_{ik})$$

and

$$\Sigma_{iyr} = Cov(Y_i, \mathbf{X}_{ir})$$
$$= (\textstyle\sum_{j=r+1}^{m} Cov(X_{ij}, X_{i1}),$$
$$..., \textstyle\sum_{j=r+1}^{m} Cov(X_{ij}, X_{ir})).$$

We define the *probing score* as

$$Sp_i \triangleq \mathbb{E}(\tau_i|\mathbf{X}_{ir}) + \lambda Sd(\tau_i|\mathbf{X}_{ir}), \qquad \text{(IV.3)}$$

where λ is the tuning parameter, and $Sd(\tau_i|\mathbf{X}_{ir}) = \sqrt{\mathrm{Var}(\tau_i|\mathbf{X}_{ir})}$. We use the probing scores decide the order of data inspection: i.e., at each iteration, the algorithm probes the machine (unit, case) with the highest probing score unless the machine has been completely probed. *The probing score can also be regarded as the upper bound of the confidence interval of the overall score, given the information gained from components that have been probed*. The algorithmic parameter λ we introduce in the probing score Sp should be chosen using training data and can serve to optimize the number of iterations..

C. A probabilistic stopping criterion for MPro

Recall that $\mathbf{X}_{ir}$ denotes the components of machine i that have been probed. We sort the machines ascending according to $\mathbb{E}(\tau_i|\mathbf{X}_{ir})$, and let $W_i = \tau_{[i]}|\mathbf{X}_{[ir]}$ denote the conditional overall score with i^{th} smallest mean. Note that W_i is not an order statistic. Let $Y = \min_{i=n-k+1}^{n} W_i$, and $Z = \max_{i=1}^{n-k} W_i$.

The stopping criterion of the algorithm is

$$\mathbb{P}(Y \ge Z) \ge \alpha \qquad \text{(IV.4)}$$

where α is the pre-determined threshold.

To calculate $\mathbb{P}(Y \ge Z)$, we use $\mathbb{P}(Y \ge Z) = \int_{-\infty}^{\infty}\int_{z}^{\infty} f_y(y)f_z(z)dzdy$, where

$$f_Y(y) = \left(\sum_{i=n-k+1}^{n} \frac{f_i(y)}{\bar{F}_i(y)}\right) \prod_{i=n-k+1}^{n} \bar{F}_i(y), \qquad \text{(IV.5)}$$

$$f_Z(z) = \left(\sum_{i=1}^{n-k} \frac{f_i(z)}{F_i(y)}\right) \prod_{i=1}^{n-k} F_i(y). \qquad \text{(IV.6)}$$

Here $F_i, \bar{F}_i$, and f_i are the c.d.f, survival function and p.d.f. of W_i, respectively. In specific, when W_i is a fixed number, that is, if machine i is fully probed, f_i is the Radon-Nikodym derivative of F_i [see (26)].

D. An MPro probabilistic algorithm

Algorithm

1) Calculate the probing scores $Sp^{(\mathcal{Q})}$ and put them in the probing queue $Q.top$, where $\mathcal{Q}$ denote the observed information.
2) Probe $Q.top$, update $\mathcal{Q}$.
3) While the stopping criterion is not satisfied, repeat step 1 and 2.

Table III
PROBABILISTIC MPRO ALGORITHM

E. Properties of the algorithm

The augmentation of the MPro algorithm with the stopping criteria for explicitly probabilistic data allows us to demonstrate that the ranking algorithm can stop as the number of iterations increases.

Theorem IV.1. Define $P_{stop} = \mathbb{P}(Y \geq Z)$, and let n_{pr} denote the total number of probes. Then $P_{stop} \to 1$ as $n_{pr} \to n \cdot m$.

Proof: When $n_{pr} = n \cdot m$, all the components of all the machines are probed, and all the conditional overall scores are fixed values. Recall that W_i's are conditional overall scores sorted ascending according to their means, then we have $\mathbb{P}(Y \geq Z) = \mathbb{P}(\min_{i=n-k+1}^{n} W_i \geq \max_{i=1}^{n-k} W_i) = 1$. Thus when $n_{pr} \to n \cdot m$, the algorithm stops almost surely. ■

As well, the choice of score — using the first and second conditional moments of τ at each iteration — is sufficient for the stopping criteria.

Theorem IV.2. The probing scores Sp are 'sufficient' to calculate P_{stop}.

Proof: Y, Z are order statistics of the conditional overall scores. Since the scores are normally distributed, the conditional overall scores also follow the normal distribution. Recall that the probing score Sp is linear combination of conditional expectation and conditional standard deviation. The Sp's are 'sufficient' to calculate P_{stop} in the sense that mean and s.d. are sufficient to characterize normal distribution. ■

V. Simulation Study

We used simulated versions of data $\mathbf{X}$ to evaluate the probabilistic augmentations of the TA and MPro algorithms: $rvTA$ and $rvMPro$. We simulated the data as multivariate normal, using parameters of real degradation data. For each top-k identification algorithm, we adopted two performance benchmarks: the probe rate and the correct rate.

The probe rate r_{probe} we define as:

$$r_{probe} = \frac{\text{\# of probes}}{n \cdot m} \tag{V.1}$$

which describes the portion of the dataset to be probed. The correct rate $r_{correct}$ we defined as

$$r_{correct} = \frac{\#\{\text{top-k machines identified correctly}\}}{k} \tag{V.2}$$

which provides a metric to measure the correctness of the algorithm.

A. TA-based algorithm evaluation

Table IV compares the performance of the TA and rvTA algorithms on these 'parameters' of the algorithm: varying fleet sizes n, component number m, and size of top-k best set k. Generally the rvTA algorithm has a slightly lesser probe length at high levels of correctness. The lack of a large reduction in probe length is due to a low tolerance level $\alpha = .05$: for higher tolerance levels, e.g. less 'cost' to error of missing some of the best/worst units (cases, machines) the rvTA algorithm outperforms the TA procedure.

For each parameter setting, we simulate the data and run the ranking algorithm 100 times; the values of r_{probe} and $r_{correct}$ in the tables and figures are averages for each of these 100 runs. The effects of different parameters r_{probe} and $r_{correct}$ are illustrated in Figures 1,2,3,4.

Table IV
THE COMPARISON OF PERFORMANCE BETWEEN TA AND RVTA

n	k	m	*TA*		*rvTA*	
			r_{probe}	$r_{correct}$	r_{probe}	$r_{correct}$
10	2	2	.346	1	.351	.970
10	2	3	.422	1	.440	.975
50	2	2	.117	1	.114	.965
50	5	2	.234	1	.221	.956
50	10	2	.381	1	.364	.951
100	2	2	.065	1	.064	.940

B. MPro-based algorithm evaluation

The MPro algorithm introduces an additional, tuning, parameter λ — the contribution of the conditional variance.

Table V
THE COMPARISON BETWEEN TWO TOP-K IDENTIFICATION ALGORITHMS, $\lambda = 2$

n	k	m	*MPro*		*rvMPro*	
			r_{probe}	$r_{correct}$	r_{probe}	$r_{correct}$
10	2	2	.805	1	.460	.930
10	2	5	.700	1	.338	.935
50	2	2	.451	1	.214	.935
50	2	5	.344	1	.143	.910
50	10	2	.741	1	.577	.986
50	10	5	.669	1	.440	.971

We notice that as n grows large, the probing rate also grows slightly, but the correctness also improves. As m increases, the probing rate also drops, yet the correctness remains at the same level. Both the $r_{probing}$ and $r_{correct}$ increase when k raises. The probe rate increases then drops as λ increases from zero — which is also true for correct rate. In practice the optimal λ could be chosen using partial data (cross validation, perhaps); we recommend , as a rule of thumb, to choose $\lambda = 2$ when the scores are normally distributed.

Table VI
THE ALGORITHM PERFORMANCE IN DIFFERENT PARAMETER SETTINGS

n	k	m	λ	r_{probe}	$r_{correct}$
10	2	2	0	.435	.925
10	2	2	1	.475	.955
10	2	2	2	.460	.930
10	2	2	3	.470	.930
10	2	5	0	.350	.935
10	2	5	1	.354	.900
10	2	5	2	.338	.935
10	2	5	3	.376	.945
50	2	2	0	.236	.930
50	2	2	1	.218	.930
50	2	2	2	.214	.935
50	2	2	3	.227	.913
50	2	5	0	.179	.910
50	2	5	1	.152	.910
50	2	5	2	.143	.910
50	2	5	3	.155	.945
50	10	2	0	.608	.984
50	10	2	1	.583	.983
50	10	2	2	.577	.986
50	10	2	3	.599	.986
50	10	5	0	.532	.981
50	10	5	1	.451	.988
50	10	5	2	.440	.971
50	10	5	3	.467	.986

C. Comparison with deterministic algorithms

Although the deterministic versions of the algorithms can always identify the top-k units correctly for TA we notice that it almost always needs to probe a larger portion of the data. For MPro, though, the probabilistic version always concludes in less, sometimes dramatically fewer, iterations. Generally speaking, as long as false identification of top-k is not fatal, the r.v.-based versions of the algorithms can be adopted. This saves the probing 'cost' at the expense of a (in some cases negligibly) lower correct rate. See Figures 5, 6, 7.

VI. SORTING MULTICOMPONENT (MULTIVARIATE) DEPENDENT DATA, MORE GENERALLY.

Without loss of generality let $\mathbf{X}$ be a non-negative multivariate random variable: the process from which $\mathbf{x}$ is drawn, from n machines (units, cases), each having m components (questions, items, etc). We can demonstrate that *the bounds for probabilistic search and scoring algorithms are set by the extremal cases of across component dependency and are measurable by the distribution of the multivariate order statistics.*

Theorem VI.1. The bounds for probabilistic search and scoring algorithms are set by the extremal cases of across component dependency and are measurable by the distribution of the multivariate order statistics.

Proof:

Let $\mathcal{F} = (\mathcal{F}_j)_{j=1...M}$ be the family of distributions for all machines; let each $\mathcal{F}_j$ be a family of univariate distributions indexed by Δt: say $\mathcal{F}_j = (F_j^{\Delta t_1}, ..., F_j^{\Delta t_N})$ such that $\overline{F}^{\Delta t_1}(x) > \overline{F}^{\Delta t_2}(x)$, $\forall x$ when $\Delta t_1 > \Delta t_2$, in concert with (8), with $\overline{F}_j^{\Delta t_i}$ the residual life distribution for component j of (ordered) machine i.

Let $\mathbf{z}$ be the ordered realizations from $\mathbf{x}$. These $\mathbf{z}$ are not order statistics in the traditional sense since while $\mathbf{x}_{i,\cdot} \perp \mathbf{x}_{i^*,\cdot}$, $\mathbf{x}$ is drawn from the *family* of distributions $\mathcal{F}$. So z_{ij} is drawn from Z_{ij} which is distributed $F^{\Delta t_i}$.

Then, since $Z_{\cdot,j}$ is stochastically increasing in i, as well $\mathbf{Z}_{i,\cdot}$ is *stochastically larger* than $\mathbf{Z}_{i^*,\cdot}$ whenever $i < i^*$ (22).

Let $T(\cdot)$ be 'scoring' or 'sorting' function from $\mathbb{R}^{M,+}$ to $\mathbb{R}^+$. Let $T_i = T(\mathbf{x}_{i,\cdot})$, $T_{z_i} = T(\mathbf{z}_{i,\cdot})$ and then $(T_{(i)})_{i=1..n}$ are the order statistics for the T_i's and $(T_{z_{(i)}})_{i=1..n}$ are the order statistics for the T_{z_i}'s. So $T_{z_{(1)}}$ is the 'score' of the stochastically largest (possible) machine while $T_{(1)}$ is the maximum observed 'score'; $T_{z_{(N)}}$ is the 'score' of the stochastically smallest (possible) machine and $T_{(N)}$ is the minimum observed 'score'.

Essentially, the sorting algorithm in (8) compares T_{z_i} with $T_{z_{(i)}}$ — both are $\mathcal{F}$ measurable but $T_{z_{(i)}} \in \mathcal{F}^{\Delta t_{i,1},...,\Delta t_{i,M}} = (F_1^{\Delta t_i}, ..., F_M^{\Delta t_i})$ whereas $T_{(i)} \in \mathcal{F} \supset \mathcal{F}^{\Delta t_{i,1},...,\Delta t_{i,M}}$.

To ask for what conditions will statistics on $\mathcal{F}^{\Delta t_{i,1},...,\Delta t_{i,M}}$ yield inference on $\mathcal{F}$ and vice versa is to say: what distributional results can be gleaned from the data for each machine without having to complete the system wide search and, vice-versa, from distributional results how can the machines be probabilistically sorted.

Let C and $\overline{C}$ be a copula and copula survival function (see (19)) on $\mathcal{F}$, s.t. each maps $\mathbb{I}^M$ to $\mathbb{I}$; C is familiar for the extremal assumptions about degradation (and thus multivariate/component dependency):

Series

$$C^{\uparrow} = C(\mathcal{F}) = \bigwedge_{j=1}^{m} \mathcal{F}_j$$

Parallel

$$C^{\downarrow} = C(\mathcal{F}) = \bigvee_{j=1}^{m} \mathcal{F}_j$$

We can call $C^{\uparrow}$ and $C^{\downarrow}$, colloquially, dependency structures where the machines are *completely weakened by failure* and *not weakened by failure*.

We are interested in the distribution of the 'scores'

$$\sigma_C(\mathcal{F})(t) = \int \cdots \int_{T(\mathbf{x})<t} dC(\mathcal{F}) \qquad \text{(VI.1)}$$

under different assumptions about the 'frailty' of the machines. Frank et al. (7) (to pick one reference) demonstrate these bounds

$$\sup_{T(\mathbf{x})=t} C(\mathcal{F}) \leq \sigma_C(\mathcal{F})(t) \leq \inf_{T(\mathbf{x})=t} \overline{C}(\mathcal{F}) \qquad \text{(VI.2)}$$

which is

$$\begin{aligned} \sup_{T(\mathbf{x})=t} C(\mathcal{F}_1, ..., \mathcal{F}_M) &\leq \sigma_C(\mathcal{F})(t) \\ &\leq \inf_{T(\mathbf{x})=t} \overline{C}(\mathcal{F}_1, ..., \mathcal{F}_M) \end{aligned} \qquad \text{(VI.3)}$$

But $\mathcal{F}_j$ is indexed, so for $T(\cdot)$ increasing

$$\begin{aligned} \sup_{T(\mathbf{x})=t} C(\mathcal{F}_1, ..., \mathcal{F}_M) &\leq \sup_{T(\mathbf{x})=t} C(F_1^{\Delta t_1}, ..., F_M^{\Delta t_1}) \\ &\leq \sigma_C(\mathcal{F})(t) \leq \inf_{T(\mathbf{x})=t} \overline{C}(F_1^{\Delta t_N}, ..., F_M^{\Delta t_N}) \\ &\leq \inf_{T(\mathbf{x})=t} \overline{C}(\mathcal{F}_1, ..., \mathcal{F}_M) \end{aligned} \qquad \text{(VI.4)}$$

But the bounds in the middle inequality are $\mathcal{F}^{\Delta t_{1,1},...,\Delta t_{1,m}}$ and $\mathcal{F}^{\Delta t_{n,1},...,\Delta t_{n,m}}$ measurable, respectively — i.e. measureable on the row-wise extremal order statistics. Note lastly that

$$\begin{aligned} C^{\uparrow} = \sup_{T(\mathbf{x})=t} C(F_1^{\Delta t_1}, ..., F_m^{\Delta t_1}) &\leq \sigma_C(\mathcal{F})(t) \\ &\leq \inf_{T(\mathbf{x})=t} \overline{C}(F_1^{\Delta t_N}, ..., F_m^{\Delta t_n}) = C^{\downarrow} \end{aligned}$$

This suggests that any type of frailty or inter-machine (unit, case) dependency is 'captured', with respect to the method of evaluation $T(\cdot)$, by the order statistics of the completely weakened by failure and not weakened by failure regimes. This is: the distribution of the ordering statistic — the sorting score — $T(\cdot) = (T_1, ..., T_N)$ is bound by the distributions of the stochastically extremal machines. *A fortiori*, we have bounds on a doubly stochastic measure via a collection of singly stochastic distributions. ■

Notice that this is a result that holds even if the machines are not identically distributed.

A. Conclusion

We offer a methodology to identify top-k among a fleet of units using sensor-based prognostic information, in particular, but the method can be immediately applied to any situation where the best/worst k of n multivariate observations are desired. Both the TA and MPro algorithms rely on sorting the multivariate data: partial in the MPro algorithm and complete for TA. Exploiting probabilistic information — i.e. explicitly assuming the data are instantiations of random variables — allows us to improve on the ordinary deterministic versions by using the within machine (unit, case) dependency. Exploiting the ranking methodology under probabilistic assumptions can significantly reduce the cost of data-acquisition.

In particular, we have derived the joint distribution of the data generated by the Brownian motion degradation model for prognostic data from (9). Using distributional information as additional input, our versions of the TA and MPro algorithms return the top-k result with only a small portion of the data. This innovation requires less sensor probes — costly for prognostic maintenance data — than the existing versions of the algorithms. When we consider correctness or tolerance level (α), we also identify the top-k units with correct rate higher than 90%.

The approach proposed in this paper has several immediate useful extensions. Here we have applied the algorithm to find the top-k machines. In fact, the method can be applied to identify either the "top" or the "bottom" units. To find the worst k units, we would merely add an negative sign to the scores, and the top-k result for the transformed variables are the bottom-k for the original scores. Lastly, when there are errors associated the estimates of the parameters $\mu_0, \mu_1, \sigma_0, \sigma_1, \sigma$ — specific to either algorithm — we can adjust the distribution of the data and apply our algorithm to them. The results hold theoretically, as well, for non-identical families of distributions.

In this paper, for the $rvMPro$ algorithm, we have determined the probing schedule, i.e.,the order of components to probe, according to former experience (8). This is to say we have relied on estimates of the parameters of the multivariate (normal) distribution for $\mathbf{X}$, specifically the correlation structure. Future research can generate an optimal probing schedule by yielding good information about the multivariate dependency/correlational structure. An optimal schedule could more perfectly exploit the dependence among the critical components to further lessen data probing iterations.

Lastly we note that the methodology here has wide ranging applications in the social and economic sciences. Multivariate, cross-national indexing, in particular, relies on sorting and scoring dependent information across administrative boundaries (3). The techniques here are designed to readily return the extremal multivariate observations and can be adjusted for tolerance and sensitivity to the cost of data acquisition.

VII. Figures

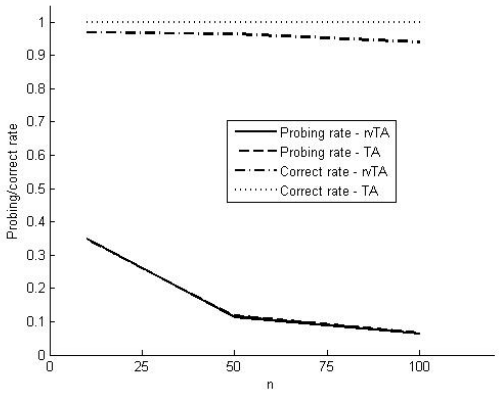

Figure 1. The comparison between rvTA and TA algorithms - Effect of n

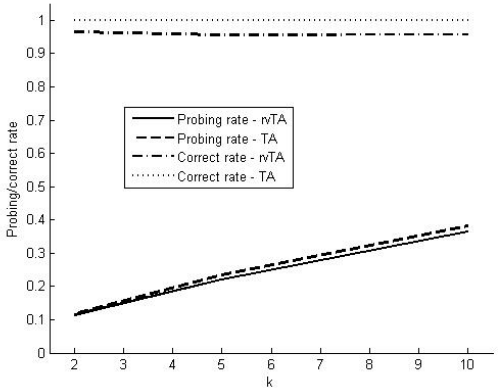

Figure 2. The comparison between rvTA and TA algorithms - Effect of k

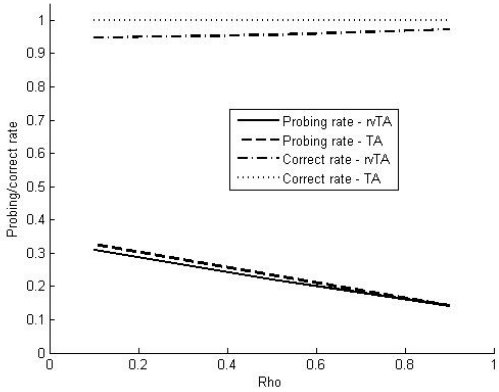

Figure 4. The comparison between rvTA and TA algorithms - Effect of ρ

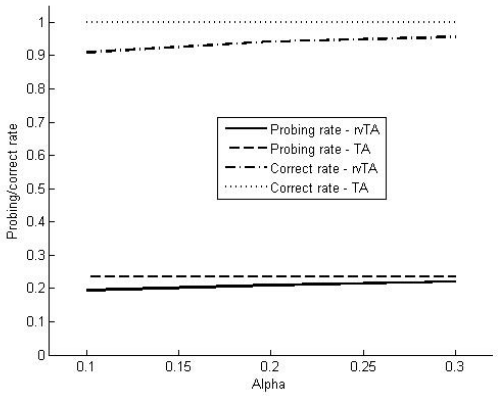

Figure 3. The comparison between rvTA and TA algorithms - Effect of α

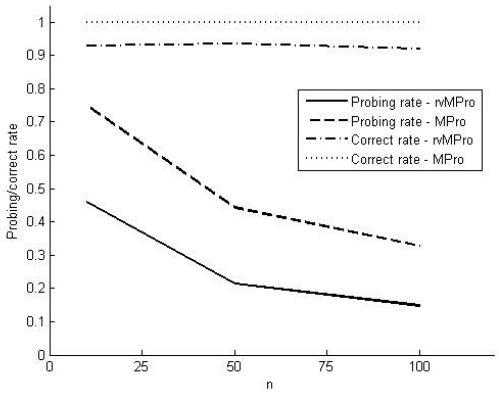

Figure 5. The comparison between two top-k identification algorithms - Effect of n

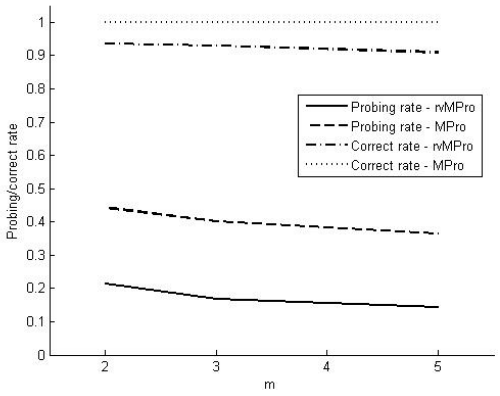

Figure 6. The comparison between two top-k identification algorithms - Effect of m

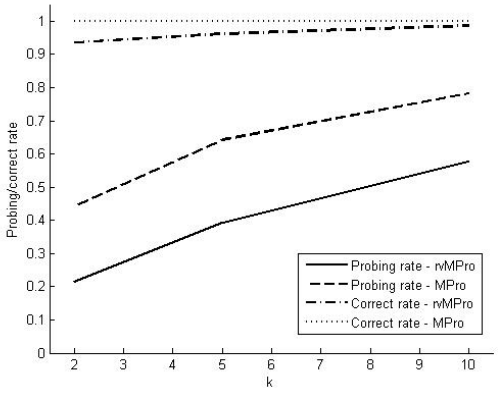

Figure 7. The comparison between two top-k identification algorithms - Effect of k

REFERENCES

[1] J. Smith C. Li A. Natsev, Y. Chang and J. Vitter, *Supporting incremental join queries on ranked inputs*, VLDB 2001 (2001), 281–290.

[2] C. Ellis K. Munagala J. Yang A. Silberstein, R. Braynard, *A sampling-based approach to optimizing top-k queries in sensor networks*, ICDE 2006 (2006), 68–78.

[3] Kobi Abayomi and Gonzalo Pizarro, *Monitoring the united nations millennium development goals: A straightforward (bayesian) methodology for a cross-national index*, Journal of Social Indicators (to appear).

[4] D. Suciu C. Re, N. Dalvi, *Efficient top-k query evaluation on probabilistic data*, ICDE 2007 (2007), 886 – 895.

[5] M. Soliman F. Ilyas, G. Beskales, *A survey of top-k query processing techniques in relational database systems*, ACM Computing Surveys **40** (2008), no. 4, 1–58.

[6] R. Fagin, *Combining fuzzy information from multiple systems*, PODS 1996 (1996), 216–226.

[7] M.J. Frank, R.B. Nelsen, and B. Schweizer, *Best possible bounds for the distribution of a sum - a problem of kolmogorov*, Probability Theory and Related Fields **74** (1987), 199–211.

[8] N Gebraeel, *Prognostics-based identification of the top-k units in a fleet*, Ieee Transactions on Automation Science and Engineering **7** (2010), no. 1, 37–48.

[9] NZ Gebraeel, MA Lawley, R Li, and JK Ryan, *Residual-life distributions from component degradation signals: A bayesian approach*, IIE Transactions **37** (2005), no. 6, 543–557.

[10] Andrew Gelman, John Carlin, Hal Stern, and Donald Rubin, *Bayesian data analysis*, Chapman & Hall/CRC, 2004.

[11] G. Kollios D. Srivastava K. Yi, F. Li, *Efficient processing of top-k queries in uncertain databases with x-relations*, IEEE transactions on knowledge and data engineering **20** (2009), no. 12, 1669–1682.

[12] S. Hwang KC. Chang, *Minimal probing: Sup-*

porting expensive predicates for topk queries, SIGMOD 2002 (2002), 346 – 357.

[13] C. Lu and W Meeker, *Using degradation measures to estimate a time-to-failure distribution*, Technometrics **35** (1993), 161–174.

[14] KC. Chang M. Soliman, I. Ilyas, *Top-k query processing in uncertain databases*, ICDE 2007 (2007), 896 – 905.

[15] R. Schenkel M. Theobald, G. Weikum, *Top-k query evaluation with probabilistic guarantees*, VLDB 2004 **30** (2004), 648–659.

[16] W. Lee D. Lee M. Ye, X. Liu, *Probabilistic top-k query processing in distributed sensor networks*, ICDE 2010 (2010), 585–588.

[17] Albert W. Marshall and Ingram Olkin, *Inequalities: Theory of majorization and its applications*, Academic Press, 1979.

[18] A. Elwany N. Gebraeel and J. Pan, *Residual life predictions in the absence of prior degradation knowledge*, IEEE TRANSACTIONS ON RELIABILITY **58** (2009), no. 1, 106–117.

[19] Roger Nelsen, *An introduction to copulas*, Springer, 2006.

[20] S. Nepal and M. Ramakrishna, *Query processing issues in image(multimedia) databases*, ICDE 1999 (1999), 22–29.

[21] A. Lote R. Fagin and M. Naor, *Optimal aggregation algorithms for middleware*, PODS 2001 (2001).

[22] M Shaked and J Shantikumar, *Dynamic construction and simulation of random vectors*, 1990.

[23] Y. Zhang J. Pei W. Wang W. Zhang, X. Lin, *Threshold-based probabilistic top-k dominating queries*, The VLDB Journal **19** (2010), no. 2, 283–305.

[24] W Wang, *A model to determine the optimal critical level and the monitoring intervals in condition-based maintenance*, International Journal of Production Research **38** (2000), 1425–1436.

[25] G. Whitmore, *Estimating degradation by a wiener diffusion process subject to measurement error*, Lifetime Data Analysis **1** (1995), 307–319.

[26] David Williams, *Probability with martingales*, Cambridge Mathematical Textbooks, Cambridge University Press, 1991.