
Kolmogorov Approximation

Anonymous Author(s)

Affiliation

Address

email

1 Introduction

Many different approaches to approximation of probability distributions are studied in the literature [2, 4]. Typically, a continuous distribution is approximated by a discrete one, but approximation is also needed when a discrete distribution is given as a large data-set, obtained, e.g., by experimentation, and we want to represent it approximately with a small table [5].

One of the most cited notion of the distance between distributions is often considered to be the distances between the corresponding commutative distribution functions (cdf). One of the most widely known distances is the, so called, Kolmogorov-Smirnov distance, which leads to the corresponding goodness of fit test, see for instance [?] and [?]. This distance is based on a single point where the absolute difference between two cdfs is maximized, and equals to the corresponding value of the absolute difference.

approximations are sometimes employed.

In this work, motivated by the problem of estimating the probability of meeting deadlines, we focus on the Kolmogorov distance $d_k(X, X') = \sup_t |F_X(t) - F_{X'}(t)|$ where F_X and $F_{X'}$ are the CDFs of X and X' , respectively.

Definition 1. A random variable X' is an m -optimal-approximation of a random variable X if $|\text{support}(X')| \leq m$ and there is no random variable X'' such that $|\text{support}(X'')| \leq m$ and $d_k(X, X'') < d_k(X, X')$.

2 An Algorithm for Optimal Approximation

- We now start our story: Given X and m how can we find X' ?
- We first show that it is enough to limit our search to X' 's such that $\text{support}(X') \subseteq \text{support}(X)$.

Lemma 2. For any discrete random variable X and any $m \in \mathbb{N}$, there is an m -optimal-approximation X' of X such that $\text{support}(X') \subseteq \text{support}(X)$.

Proof. Assume there is a random variable X'' with support size m such that $d_K(X, X'')$ is minimal but $\text{support}(X'') \not\subseteq \text{support}(X)$. We will show how to transform X'' support such that it will be contained in $\text{support}(X)$. Let v' be the first $v' \in \text{support}(X'')$ and $v' \notin \text{support}(X)$. Let $v = \max\{i : i < v' \wedge i \in \text{support}(X)\}$. Every v' we will replace with v and name the new random

variable X' , we will show that $d_K(X, X'') = d_K(X, X')$. First, note that: $F_{X''}(v') = F_{X'}(v)$,
 $F_X(v') = F_X(v)$. Second, $F_{X'}(v') - F_X(v') = F_{X'}(v) - F_X(v)$. Therefore, $d_K(X, X'') =$
 $d_K(X, X')$ and X' is also an optimal approximation of X . \square

Observation 3. $\max\{|a|, |b|\} \geq |a - b|/2$

• The next lemma states a lower bound on the distance $d_K(X, X')$ when a range of elements
is excluded from the support of X' .

Lemma 4. For $x_1, x_2 \in \text{support}(X) \cup \{-\infty, \infty\}$ such that $x_1 < x_2$, if $P(x_1 < X' < x_2) = 0$
then $d_k(X, X') \geq P(x_1 < X < x_2)/2$.

Proof. Let $\hat{x} = \max\{x \in \text{support}(X) \cap \{-\infty, \infty\} : x < x_2\}$. By definition, $d_k(X, X') \geq$
 $\max\{|F_X(x_1) - F_{X'}(x_1)|, |F_X(\hat{x}) - F_{X'}(\hat{x})|\}$. From Observation 3, $d_k(X, X') \geq 1/2|F_X(x_1) -$
 $F_X(\hat{x}) + F_{X'}(\hat{x}) - F_{X'}(x_1)|$. Since it is given that $F_{X'}(\hat{x}) - F_{X'}(x_1) = P(x_1 < X' < x_2) = 0$,
 $d_k(X, X') \geq 1/2|F_X(x_1) - F_X(\hat{x})| = P(x_1 < X \leq \hat{x})/2 = P(x_1 < X < x_2)/2$. \square

• The next lemma strengthen the lower bound.

Lemma 5. For $x_1, x_2 \in \text{support}(X) \cup \{-\infty, \infty\}$ such that $x_1 = -\infty$ or $x_2 = \infty$, if $P(x_1 <$
 $X' < x_2) = 0$ then $d_k(X, X') \geq P(x_1 < X < x_2)$.

Proof. Let $\hat{x} = \max\{x \in \text{support}(X) \cap \{-\infty, \infty\} : x < x_2\}$. By definition $d_k(X, X') \geq$
 $\max\{|F_X(x_1) - F_{X'}(x_1)|, |F_X(\hat{x}) - F_{X'}(\hat{x})|\}$. If $x_1 = -\infty$ then $d_k(X, X') \geq \{|F_X(\hat{x}) -$
 $F_{X'}(\hat{x})|\}$ since $F_X(-\infty) = F_{X'}(-\infty) = 0$. Furthermore, $F_{X'}(\hat{x}) = P(x_1 < X' < x_2) =$
 0 . Therefore $d_k(X, X') \geq F_X(\hat{x}) = P(x_1 < X \leq \hat{x}) = P(x_1 < X < x_2)$. If $x_2 = \infty$
then $d_k(X, X') \geq \{|F_X(x_1) - F_{X'}(x_1)|\}$ since $F_X(\hat{x}) = F_{X'}(\hat{x}) = F_X(\infty) = F_{X'}(\infty) = 1$.
Furthermore, $F_{X'}(x_1) = 1$ since it is given that $P(x_1 < X' < x_2) = 0$. Therefore we get that
 $d_k(X, X') \geq |F_X(x_1) - 1| = |1 - F_X(\hat{x})| = P(x_1 < X \leq \hat{x}) = P(x_1 < X < x_2)$. \square

Definition 6. For $x_1, x_2 \in \text{support}(X) \cup \{-\infty, \infty\}$ let

$$w(x_1, x_2) = \begin{cases} P(x_1 < X < x_2) & \text{if } x_1 = -\infty \text{ or } x_2 = \infty; \\ P(x_1 < X < x_2)/2 & \text{otherwise.} \end{cases}$$

Definition 7. For $S = \{x_1 < \dots < x_m\} \subseteq \text{support}(X)$, $x_0 = -\infty$, and $x_{m+1} = \infty$, let

$$\varepsilon(X, S) = \max_{i=0, \dots, m} w(x_i, x_{i+1}).$$

• From here on, until the end of the section, S is fixed.

Proposition 8. There is no X' such that $\text{support}(X') = S$ and $d_k(X, X') < \varepsilon(X, S)$.

Proof. Let i be the index that maximizes $w(x_i, x_{i+1})$. If $0 < i < n - 1$ then $d_k(X, X') \geq$
 $w(x_i, x_{i+1})$ by Lemma 4. If $i = 0$ or $i = n + 1$ the same follows from Lemma 5. \square

Definition 9. Let X' to be $f_{X'}(x_i) = w(x_{i-1}, x_i) + w(x_i, x_{i+1}) + f_X(x_i)$ for $i = 1, \dots, m$ and
 $f_{X'}(x) = 0$ for $x \notin S$.

Lemma 10. For $i > 1$, if $F_{X'}(x_i) - F_X(x_i) = w(x_i, x_{i+1})$ then $F_{X'}(x_{i+1}) - F_X(x_{i+1}) =$
 $w(x_{i+1}, x_{i+2})$.

Proof.

$$F_{X'}(x_{i+1}) - F_X(x_{i+1}) = \quad (1)$$

$$\begin{aligned} &= f_{X'}(x_{i+1}) - f_X(x_{i+1}) - P(X < x_{i+1}) + P(X' < x_{i+1}) \\ &= f_{X'}(x_{i+1}) - f_X(x_{i+1}) - F_X(x_i) - P(x_i < X < x_{i+1}) + F_{X'}(x_i) \\ &= f_{X'}(x_{i+1}) - f_X(x_{i+1}) - F_X(x_i) - 2w(x_i, x_{i+1}) + F_{X'}(x_i) \end{aligned} \quad (2)$$

$$= f_{X'}(x_{i+1}) - f_X(x_{i+1}) - 2w(x_i, x_{i+1}) + w(x_i, x_{i+1}) \quad (3)$$

$$= w(x_i, x_{i+1}) + w(x_{i+1}, x_{i+2}) - 2w(x_i, x_{i+1}) + w(x_i, x_{i+1}) \quad (4)$$

$$= w(x_{i+1}, x_{i+2})$$

61 By Definition 6 the probability $P(x_{i-1} < X < x_i) = 2w(x_{i-1}, x_i)$ as in Equation (2). Equation (3)
 62 is deduced by the induction hypothesis and Equation (4) where $f_{X'}(x_i) - f_X(x_i) = w(x_{i-1}, x_i) +$
 63 $w(x_i, x_{i+1})$ is true by construction, see Definition 9. \square

64 **Lemma 11.** *Base case:* $i = 1, F_{X'}(x_1) - F_X(x_1) = w(x_1, x_2)$.

Proof.

$$F_{X'}(x_1) - F_X(x_1) =$$

$$\begin{aligned} &= f_{X'}(x_1) - f_X(x_1) - w(x_0, x_1) \\ &= w(x_0, x_1) + w(x_1, x_2) - w(x_0, x_1) \\ &= w(x_1, x_2) \end{aligned}$$

65 \square

66 **Proposition 12.** *There exists X' such that $\text{support}(X') = S$ and $d_k(X, X') = \varepsilon(X, S)$.*

67 Chakravarty, Orlin, and Rothblum [1] proposed a polynomial-time method that, given certain objective
 68 functions (additive), finds an optimal consecutive partition. Their method involves the construction
 69 of a graph such that the (consecutive) set partitioning problem is reduced to the problem of finding
 70 the shortest path in that graph.

71 The KolmogorovApprox algorithm (Algorithm 2) starts by constructing a directed weighted graph
 72 G similar to the method of Chakravarty, Orlin, and Rothblum [1]. The nodes V consist of the
 73 support of X together with an extra two nodes ∞ and $-\infty$ for technical reasons, whereas the
 74 edges E connect every pair of nodes in one direction (lines 1-2). The weight w of each edge
 75 $e = (i, j) \in E$ is determined by one of two cases. The first is where i or j are the source or target
 76 nodes respectively. In this case the weight is the probability of X to get a value between i and
 77 j , non inclusive, i.e., $w(e) = \Pr(i < X < j)$ (lines 4-5). The second case is where i or j are
 78 not a source or target nodes, here the weight is the probability of X to get a value between i and
 79 j , non inclusive, divided by two i.e., $w(e) = \Pr(i < X < j)/2$ (lines 6-7). The values taken
 80 are non inclusive, since we are interested only in the error value. The source node of the shortest
 81 path problem at hand corresponds to the $-\infty$ node added to G in the construction phase, and the
 82 target node is the extra node ∞ . The set of all solution paths in G , i.e., those starting at $-\infty$ and
 83 ending in ∞ with at most m edges, is called $\text{paths}(G, -\infty, \infty)$. The goal is to find the path l^*
 84 in $\text{paths}(G, -\infty, \infty)$ with the lightest bottleneck (lines 8-9). This can be achieved by using the
 85 Bellman – Ford algorithm with two tweaks. The first is to iterate the graph G in order to find only
 86 paths with length of at most m edges. The second is to find the lightest bottleneck as opposed to

87 the traditional objective of finding the shortest path. This is performed by modifying the manner of
 88 “relaxation” to $bottleneck(x) = \min[\max(bottleneck(v), w(e))]$, done also in [6]. Consequently,
 89 we find the lightest maximal edge in a path of length $\leq m$, which represents the minimal error, ε^* ,
 90 defined in Definition ?? . X' is then derived from the resulting path l^* (lines 10-17). Every node
 91 $n \in l^*$ represent a value in the new calculated random variable X' , we than iterate the path l^* to fine
 92 the probability of the event $f_{X'}(n)$. For every edge $(i, j) \in l^*$ we determine: if (i, j) is the first edge
 93 in the path l^* (i.e. $i == -\infty$), then node j gets the full weight $w(i, j)$ and it's own weight in X
 94 such that $f_{X'}(j) = f_X(j) + w(i, j)$ (lines 11-12). If (i, j) in not the first nor the last edge in path
 95 l^* then we divide it's weight between nodes i and j in addition to their own original weight in X
 96 and the probability that already accumulated (lines 16-17). If (i, j) is the last edge in the path l^* (i.e.
 97 $i == \infty$) then node i gets the full weight $w(i, j)$ in addition to what was already accumulated such
 98 that $f_{X'}(j) = f_{X'}(j) + w(i, j)$ (lines 13-14).

Algorithm 1: KolmogorovApprox(X, m)

```

1  $S = \text{support}(X) \cup \{\infty, -\infty\}$ 
2  $G = (V, E) = (S, \{e = (x, y) \in S^2 : x < y\})$ 
3  $l = \text{argmin}_{l \in \text{paths}(G, -\infty, \infty), |l| \leq m} \max\{w(e) : e \in l\}$ 
4 for  $0 < i < m + 1$  do
5    $f_{X'}(x_i) = w(x_{i-1}, x_i) + w(x_i, x_{i+1}) + f_X(x_i)$ 
6 return  $X'$ 

```

Algorithm 2: KolmogorovApprox(X, m)

```

1  $S = \text{support}(X) \cup \{\infty, -\infty\}$ 
2  $G = (V, E) = (S, \{e = (x, y) \in S^2 : x < y\})$ 
3 foreach  $e = (x, y) \in E$  do
4   if  $i = \infty$  OR  $j = -\infty$  then
5      $w(e) = \text{Pr}(i < X < j)$ 
6   else
7      $w(e) = \text{Pr}(i < X < j)/2$ 
8 /* The following can be obtained, e.g., using the Bellman-Ford algorithm */
9  $l^* = \text{argmin}_{l \in \text{paths}(G, -\infty, \infty), |l| \leq m} \max\{w(e) : e \in l\}$ 
10 foreach  $e = (i, j) \in l^*$  do
11   if  $i = -\infty$  then
12      $f_{X'}(j) = f_X(j) + \text{Pr}(i \leq X < j)$ 
13   else if  $j == \infty$  then
14      $f_{X'}(i) = f_{X'}(i) + \text{Pr}(i \leq X < j)$ 
15   else
16      $f_{X'}(i) = f_{X'}(i) + \text{Pr}(i \leq X < j)/2$ 
17      $f_{X'}(j) = f_X(j) + \text{Pr}(i \leq X < j)/2$ 
18 return  $X'$ 

```

99 **Theorem 13.** The KolmogorovApprox(X, m) algorithm runs in time $O(mn^2)$, using $O(n^2)$ mem-
 100 ory where $n = |\text{support}(X)|$.

101 *Proof.* Constructing the graph G takes $O(n^2)$. The number of edges is $O(E) \approx O(n^2)$ and for every
 102 edge the weight is at most the sum of all probabilities between the source node $-\infty$ and the target
 103 node ∞ , which can be done efficiently by aggregating the weights of already calculated edges. The
 104 construction is also the only stage that requires memory allocation, specifically $O(E + V) = O(n^2)$.

105 Finding the shortest path takes $O(m(E + V)) \approx O(mn^2)$. Since G is DAG (directed acyclic graph)
 106 finding shortest path takes $O(E + V)$. We only need to find paths of length $\leq m$, which takes
 107 $O(m(E + V))$. Deriving the new random variable X' from the computed path l^* takes $O(mn)$. For
 108 every node in l^* (at most m nodes), calculating the probability $P(s < X < \infty)$ takes at most n .
 109 To conclude, the worst case run-time complexity is $O(n^2 + mn^2 + mn) = O(mn^2)$ and memory
 110 complexity is $O(E + V) = O(n^2)$. \square

111 3 Experiments and Results

112 In the first experiment we focus on the problem of task trees with deadlines, and consider three
 113 types of task trees. The first type includes logistic problems of transporting packages by trucks and
 114 airplanes (from IPC2 <http://ipc.icaps-conference.org/>). Hierarchical plans of those logistic problems
 115 were generated by the JSHOP2 planner [3] (see example problem, Figure 1). The second type consists
 116 of task trees used as execution plans for the ROBIL team entry in the DARPA robotics challenge
 117 (DRC simulation phase), and the third type is of linear plans (sequential task trees). The primitive
 118 tasks in all the trees are modeled as discrete random variables with support of size M obtained by
 119 discretization of uniform distributions over various intervals. The number of tasks in a tree is denoted
 by N .

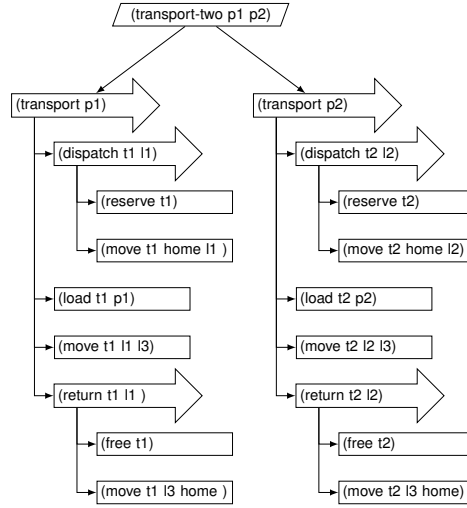


Figure 1: A plan generated by the JSHOP2 algorithm. Arrow shapes represent sequence nodes, parallelograms represent parallel nodes, and rectangles represent primitive nodes.

120

121 We implemented the approximation algorithm for solving the deadline problem with four different
 122 methods of approximation. The first two are for achieving a one-sided Kolmogorov approximation –
 123 the OptTrim and the Trim operators, and a simple sampling scheme which we used as comparison
 124 to the Kolmogorov approximation with the KolmogorovApprox algorithm. The parameter m of
 125 OptTrim and KolmogorovApprox corresponds to the inverse of ε given to the Trim operator. Note
 126 that in order to obtain some error ε , one must take into consideration the size of the task tree,
 127 N , therefore, $m/N = 1/(\varepsilon \cdot N)$. We ran the algorithm for exact computation as reference, the
 128 approximation algorithm using KolmogorovApprox as its operator with $m = 10 \cdot N$, the OptTrim
 129 as its operator with $m = 10 \cdot N$, the Trim as operator with $\varepsilon = 0.1/N$, and two simple simulations,
 130 with a different samples number $s = 10^4$ and $s = 10^6$.

Task Tree	M	OptTrim	Trim	Sampling	
		$m/N=10$	$\varepsilon \cdot N=0.1$	$s=10^4$	$s=10^6$
Logistics ($N=34$)	2	0	0.0019	0.007	0.0009
	4	0.0046	0.0068	0.0057	0.0005
Logistics ($N=45$)	2	0.0005	0.002	0.015	0.001
	4	0.003	0.004	0.008	0.0006
DRC-Drive ($N=47$)	2	0.004	0.009	0.0072	0.0009
	4	0.008	0.019	0.0075	0.0011
Sequential ($N=10$)	4	0.024	0.04	0.008	0.0016
	10	0.028	0.06	0.0117	0.001

Table 1: Comparison of estimation errors with respect to the reference exact computation on various task trees.

m	OptTrim	Trim	Relative error
2	0.491	0.493	0.4%
4	0.242	0.247	2.1%
8	0.118	0.123	4.4%
10	0.093	0.099	6%
20	0.043	0.049	15%
50	0.013	0.019	45.4%

Table 2: OptTrim vs. Trim on randomly generated random variables with original support size $M = 100$.

Table 1 shows the results of the main experiment. The quality of the solutions provided by using the OptTrim operator are better (lower errors) than those provided by the Trim operator, following the optimality guarantees, but is interesting to see that the quality gaps happen in practice in each of the examined task trees. However, in some of the task trees the sampling method produced better results than the approximation algorithm with OptTrim. Nevertheless, the approximation algorithm comes with an inherent advantage of providing an exact quality guarantees, as opposed to the probabilistic guarantees provided by sampling.

In order to better understand the quality gaps in practice between OptTrim and Trim, we investigate their relative errors when applied on single random variables with different sizes of the support (M), and different support sizes of the resulting random variable approximation (m). In each instance of this experiment, a random variable is randomly generated by choosing the probabilities of each element in the support from a uniform distribution and then normalizing these probabilities so that they sum to one.

Tables 2 and 3 present the error produced by OptTrim and Trim on random variables with supports sizes of $M = 100$ and $M = 1000$, respectively. The depicted results in these tables are averages over several instances of random variables for each entry (50 instances in Table 2 and 10 instances in Table 3). The two central columns in each table show the average error of each method, whereas the right column presents the average percentage of the relative error of the Trim operator with respect to the error of the optimal approximation provided by OptTrim; the relative error of each instance is calculated by $(\text{Trim} / \text{OptTrim}) - 1$. According to the depicted results it is evident that increasing the support size of the approximation m reduces the error, as expected, in both methods. However, the interesting phenomenon is that the relative error percentage of Trim grows with the increase of m .

The above experiments display the quality of approximation provided by the OptTrim algorithm, but it comes with a price tag in the form of run-time performance. The time complexity of both the Trim operator and the sampling method is linear in the number of variables, resulting in much faster run-time performances than OptTrim, for which the time complexity is only polynomial (Theorem 13), not linear. The run-time of the exact computation, however, may grow exponentially.

m	OptTrim	Trim	Relative error
50	0.0193	0.0199	3.4%
100	0.0093	0.0099	7.1%
200	0.0043	0.0049	15.7%

Table 3: OptTrim vs. Trim on randomly generated random variables with original support size $M = 1000$.

Therefore, we examine in the next experiment the problem sizes in which it becomes beneficial in terms of run-time to use the proposed approximation.

Figure 2 presents a comparison of the run-time performances of an exact computation and approximated computations with OptTrim and Trim as operators. The computation is a summation of a sequence of random variables with support size of $M=10$, where the number N of variables varies from 6 to 19. In this experiment, we executed the OptTrim operator with $m=10$ after performing each convolution between two random variables, in order to maintain a support size of 10 in all intermediate computations. Equivalently, we executed the Trim operator with $\varepsilon = 0.1$. The results clearly show the exponential run-time of the exact computation, caused by the convolution between two consecutive random variables. In fact, in the experiment with $N=20$, the exact computation ran out of memory. These results illuminate the advantage of the proposed OptTrim algorithm that balances between solution quality and run-time performance – while there exist other, faster, methods (e.g., Trim), OptTrim provides high-quality solutions in reasonable (polynomial) time, which is especially important when an exact computation is not feasible, due to time or memory.

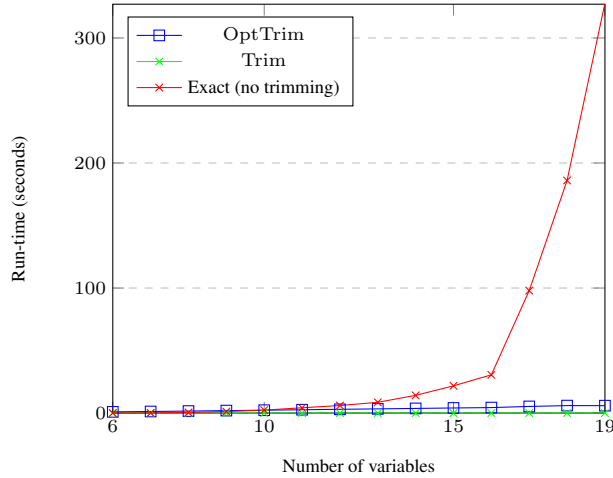


Figure 2: Run-time of a long computation with OptTrim, with Trim, and without any trimming (exact computation).

References

- [1] A. Chakravarty, J. Orlin, and U. Rothblum. A partitioning problem with additive objective with an application to optimal inventory groupings for joint replenishment. *Operations Research*, 30(5):1018–1022, 1982.
- [2] A. C. Miller and T. R. Rice. Discrete approximations of probability distributions. *Management Science*, 29(3):352–362, 1983.
- [3] D. S. Nau, T.-C. Au, O. Ilghami, U. Kuter, J. W. Murdock, D. Wu, and F. Yaman. SHOP2: An HTN planning system. *Journal of Artificial Intelligence Research*, 20:379–404, 2003.

- 181 [4] K. Pavlikov and S. Uryasev. CVaR distance between univariate probability distributions and
182 approximation problems. Technical Report 2015-6, University of Florida, 2016.
- 183 [5] A. N. Pettitt and M. A. Stephens. The kolmogorov-smirnov goodness-of-fit statistic with discrete
184 and grouped data. *Technometrics*, 19(2):205–210, 1977.
- 185 [6] E. Shufan, H. Ilani, and T. Grinshpoun. A two-campus transport problem. In *MISTA*, pages
186 173–184, 2011.