Machine learning based trajectory optimization

Alok Shukla
alok.shukla@umanitoba.ca
University of Manitoba

Prakash Vedula
pvedula@ou.edu
University of Oklahoma

Abstract

In a recent work [15], we presented a framework to transform a global trajectory optimization problem (or a problem involving calculus of variations) in the form of a search problem in a discrete space, which could then be solved by a variety of techniques, most notably, including quantum computational algorithms. In the present work we further focus on the problem of discretization and describe an improvement in our earlier proposed scheme ([15]), specially in the context of high dimensional problems. Here we offer a framework, wherein the curse of dimensionality is effectively handled by exploring and exploiting the inherent structure present in the approximate solutions obtained by probabilistic methods. Our method relies on the correlation present in the appropriately sampled discrete state space data. We use a PCA (Principal Component Analysis) based approach to efficiently perform discretization in a lower dimensional transformed space. Thereby, the size of the search space is effectively reduced. Finally, an inverse PCA transform is performed to obtain an approximate solution of the original optimization.

1 Introduction

Trajectory optimization problems or the problems related to calculus of variations are important classes of problems, not only in engineering but also in physical sciences. For example, the principle of least action in physics is based on variational principles. A variety of both the direct and the indirect methods of solving such problems are known (see [2], [5], [7], [17]). Machine learning based approaches and statistical techniques, such as Bayesian optimization, were leveraged for solving global optimization problems (see [11], [16], [18], [19]).

In the present work, we propose a new machine learning based approach to solve global optimization problems, which can be applied either in conjunction with quantum computational algorithms or with classical random or exhaustive search algorithms. We recall that a quantum computational approach to solve trajectory optimization problems was formulated in our previous work (see ([15])). The use of quantum algorithms is attractive, as these algorithms, which are essentially based on Grover's search algorithm (see [8]) offer a quadratic speed up in comparison to a classical algorithm. One of the main ingredients

Keywords: Trajectory optimization, calculus of variations, global optimization, machine learning, Principal Component Analysis, PCA, Brachistochrone problem.

in our framework in [15] was a novel discretization scheme, which made the application of efficient quantum computation algorithms possible in finding optimum trajectories. In the present work we focus on improving the discretization scheme presented in [15]. We use a PCA (Principal Component Analysis) based approach to exploit the structure present in appropriately sampled data, based on selection of the best candidate solutions analyzed so far, to pick the best sample points in the independent variables. This results in an efficient discretization scheme with an overall reduction in computational cost for solving the optimization problem. We will illustrate our method by applying it to two classical problems, including (a) the brachistochrone problem, which was also tackled in our earlier paper [15], and (b) the isoperimetric problem. Further, we will also solve a typical optimal temperature control problem using our method.

It is important to note that although PCA is a well-known method, its application in the present work for selecting sampling points for discretization is novel, especially in the context of optimization problems (and also problems related to calculus of variations). Our PCA based proposed approach facilitates discretization in a lower dimensional transformed space. The reduced size of the search space is specially helpful in tackling high dimensional global optimization problems. In fact, the use PCA based approach will allow one to solve many such high dimensional global optimization problems, which can not be solved by using a random or exhaustive search based method using the traditional (non PCA based) discretization approach. Although, the method presented in this work relies on classical computations, it is also useful in tackling trajectory optimization problems using quantum computational approaches (based on the framework presented in [15], and therefore it can), making this even more attractive.

In Sect. 2 and Sect. 3, we recall the problem formulation and our discretization scheme given in [15]. Next in Sect. 4, we provide a brief introduction to PCA. In Sect. 5, we present the main idea of this paper by describing the application of PCA in discretization. In the next section, Sect. 6, we present computational examples to illustrate the method of Sect. 5 and also evaluate the performance of the method. We present a short discussion on low rank matrix approximation methods for performing SVD (and PCA) in Sect. 7. Finally, the conclusion is presented in Sect. 8.

2 Problem formulation

The basic problem formulation is the same as in [15], which we reproduce for the convenience of the reader.

Suppose $\vec{U}(t) \in \mathbb{R}^m$ and $\vec{X}(t) \in \mathbb{R}^n$ denote the control function and the corresponding state trajectory respectively at time t. The goal is to determine the optimal control function $\vec{U}(t)$ and the corresponding state trajectory $\vec{X}(t)$ for $\tau_0 \leq t \leq \tau_f$, such that the following Bolza cost function is minimized:

$$\mathcal{J}(\vec{U}(.), \vec{X}(.), \tau_f) = \mathcal{M}(\vec{X}(\tau_f), \tau_f) + \int_{\tau_0}^{\tau_f} \mathcal{L}(\vec{U}(\tau), \vec{X}(t), t)\, dt. \tag{2.1}$$

Here $\mathcal{M}$ and $\mathcal{L}$ are $\mathbb{R}$ valued functions. Moreover, we also assume that the system satisfies

the following dynamic constraints.

$$f_l \leq f(\vec{U}(\tau), \vec{X}(t), \vec{X}'(t), t) \leq f_u \qquad t \in [\tau_0, \tau_f]. \tag{2.2}$$

In addition, we also specify the following boundary conditions

$$h_l \leq h(\vec{X}(\tau_0), \vec{X}(\tau_f), \tau_f - \tau_0) \leq h_u, \tag{2.3}$$

with h a $\mathbb{R}^p$ valued function and $h_l, h_u \in \mathbb{R}^p$ are constant vectors providing the lower and upper bounds of h. Finally we note the mixed constraints on control and state variables

$$g_l \leq g(\vec{U}(t), \vec{X}(t), t) \leq g_u, \tag{2.4}$$

with g a $\mathbb{R}^r$ valued function and $g_l, g_u \in \mathbb{R}^r$ are constant vectors providing the lower and upper bounds of g.

3 Discretization

We described our discretization scheme in [15] in detail, wherein we divided the time interval $[\tau_0, \tau_f]$ into η sub-intervals $[t_j, t_{j+1}]$ for $j = 0, 1 \cdots \eta$ with $\tau_0 = t_0 < t_1 < t_2 < \cdots < t_{\eta-1} < t_\eta = \tau_f$. Essentially, for each time instance t_i the state variable $\vec{X}(t_i)$ can be discretized with a finite number of states, which results in a finite search space (say of size N) for the optimization problem. Alternatively, for $t \in [t_j, t_{j+1}]$ the the state variable $\vec{X}(t)$ can be approximated with an appropriately chosen spectral polynomial. Owing to some interesting results in approximation theory (ref. [13], Theorems 16.3.1 and 16.3.2 therein) on the bounds on coefficients of bounded polynomials, there exists a solid mathematical foundation for carrying out discretization in the co-efficient space of these polynomials. For example, suppose $x(t)$ is a component of the state variable $\vec{X}(t)$ such that $x_0 < x(t) < x_1$ for $t \in [t_0, t_1]$. Then, $x(t)$ can be approximated by the polynomial $\sum_{n=0}^{N} b_n t^n$ for $t \in [t_0, t_1]$. From the given constraint $x_0 < x(t) < x_1$, Theorems 16.3.1 and 16.3.2,[13], provide bounds on the coefficients b_n in terms of the coefficients of Chebyshev polynomials of the first kind. These bounds can then be used to discretize the coefficient b_n (see Sect. 6.1.2, [15] for a computational example).

Once the discretization is carried out, in principle, one can then solve the problem by transversing the search space and determining the state corresponding to the minimum cost and satisfying all the imposed constraints. Moreover, the quality of the solution thus obtained can be further improved by carrying out a finer discretization resulting in a larger search space. However, the large size of the search space is often problematic in practice. For some problems this may even be totally prohibiting. We propose to use Principal Component Analysis (PCA) to address this issue and before proceeding further, we quickly recall the basic principles of PCA (in Sect. 4). We refer the readers to standard books on Multivariate Statistical Analysis for further details, for example [4], [6] and [10].

4 A brief introduction to PCA

Principal Component Analysis (PCA) is a statistical technique used for transforming a high-dimensional dataset (often called a feature space) into a smaller-dimensional subspace that

still captures most of the useful information present in the larger dataset. Suppose we have a feature space of dimension n (or equivalently, n random variables) and there are m sample observations of these features. The first principal component is the direction of the feature space along which the variance is maximum. The second principal component is orthogonal to the first principal component such that it captures the maximum variance among all the directions that are orthogonal to the first principal component. Proceeding in this way one can define all other principal components. The advantage of this approach is that one can often account for almost all the data variability by considering only k principal components, with $k < n$ and thus the original data set of m measurements on n variables (or features) is approximately captured by a dataset consisting of m measurements on k principal components. Alternatively, but equivalent to the above description, the principal components are defined to be the eigenvectors of the covariance matrix. More explicitly, let $X = [\, X_1 \;\; X_2 \;\; \cdots \;\; X_n \,]^T$ be a random vector having the covariance matrix X_V with eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq 0$. Let e_i represent the eigenvector of X_V corresponding to the eigenvalue λ_i for $i = 1, 2 \ldots n$. Then the i-th principal component is represented as

$$Z_i = e_i^T \cdot X = \sum_{j=1}^{n} e_{ij} X_j.$$

It can be shown that, for $i = 1, 2, \cdots, n$ the variances are given as

$$Var(Z_i) = e_i^T \cdot X_V \cdot e_i = \lambda_i,$$

and for $i \neq j$, the covariances are given as

$$Cov(Z_i, Z_j) = e_i^T \cdot X_V \cdot e_j = 0.$$

Thus essentially, the n-dimensional random vector X is transformed to a k-dimensional random vector $Z = [Z_1, Z_2, \cdots, Z_k]$ in what we will call the PCA transformation. We note that the total variance of the random vector X, i.e., $\sum_{i=1}^{n} Var(X_i)$ can be shown to be equal to $\sum_{i=1}^{n} \lambda_i$ (see Chapter 8, [10]). Since the i-th principal component captures

$$\frac{\lambda_i}{\sum_{j=1}^{n} \lambda_j}$$

proportion of the total population variance, the random vector Z accounts for the total

$$\frac{\sum_{j=1}^{k} \lambda_j}{\sum_{j=1}^{n} \lambda_j}$$

proportion of the total population variance. For example, if the first 3 principal components capture more than 90% of the total population variance, then (for many practical applications) it is enough to consider only these 3 principal components.

5 Application of PCA in discretization

If the size of the search space is very large, it is advisable to look for any structure present in the partial solutions. Any structure present in the data coming from partial solutions can be exploited to limit the size of the search space and thereby facilitating an efficient solution. In the following we describe our PCA based framework for solving global trajectory optimization problems.

Step I Consider each component of the state variable $\vec{X}(t_j) = [X_1(t_j), X_2(t_j), \cdots X_n(t_j)]^T$ as a discrete random variable. Let us write X_{ij} for the component $X_i(t_j)$ for the notational brevity. We treat X_{ij} as discrete random variable and we impose the condition that X_{ij} can only take values from a finite discrete set. Combining them all together, let

$$\tilde{X}_i = [\ X_{i1},\ X_{i2},\ \cdots.\ X_{i\eta}\ ]^T$$

be the random vector corresponding to the i-th component of the state variable $\vec{X}(t)$.

Step II Carry out the random search algorithm (see Sect. 4.2, [15]) for M number of times to get M sample observations for the random vector $\tilde{X}_i$, for $i = 1, 2, \cdots n$.

Step III Pick top R optimum sample observations selected based on the cost function (i.e., R observations with the lowest costs), out of the M observations determined in the previous step. Essentially, each sample observation has an associated cost, and here R samples with the lowest costs are selected.

Step IV Use the R top sample observations, and calculate the covariance matrix $Cov(\tilde{X}_i)$, for $i = 1, 2, \cdots n$.

Step V Perform PCA transformation of $\tilde{X}_i$ to get the random vector $\tilde{Z}_i = [Z_{i1}, Z_{i2}, \cdots, Z_{ik}]$ consisting of the top k principal components. Carry out discretization in the transformed space using the vector $\tilde{Z}_i$. Here a finer discretization may be considered. For carrying out discretization in the transformed space, a guiding factor could be the standard deviation in the direction of each of the principal component. For example, one can restrict each of the random variables to take values in discrete sets within a range of ± 3 standard deviations. We will further illustrate this point using computational examples in Sect. 6.

Step VI Use inverse transform to get the discretization of the random vector $\tilde{X}_i$.

Step VII Use Exhaustive or Random search as described in (see Sect. 4, [15]) to find the minimum cost.

Remark: We note that the number of initial random sample observation M depends upon the nature of the problem. Of course, the bigger the value of M, the better will be the quality of solution. But the bigger number of sample observations M, comes at an increased computational cost. Therefore, a judgment has to be made about the size of M, depending upon the nature of the problem. Similarly, the choice of R has a trade-off. A too big value

of R will possibly include data from unwanted 'bad' solutions, whereas a too small value of R may miss to capture potential 'good' solutions. We give several computational examples later in this work, along with values of M and R.

6 Computational examples

6.1 Brachistochrone problem

We apply our algorithm to solve the well-known Brachistochrone problem. The objective in the brachistochrone problem is to determine the trajectory of a particle starting from the rest under the influence of gravity, without any friction, such that it slides from the one fixed point to the other in the least possible time. Let $\vec{X}(t) = [X_1(t), X_2(t)] = x(t)\vec{i} + y(t)\vec{j}$ represents the position of the particle. Let the boundary condition be $(x(\tau_0), y(\tau_0)) = (0, 2)$ and $(x(\tau_f), y(\tau_f)) = (\pi, 0)$ with $\tau_0 = 0$. The goal is to minimize τ_f given by

$$\tau_f = \int_0^{\pi} \sqrt{\frac{1 + (\frac{dy}{dx})^2}{2gy}}\, dx. \tag{6.1}$$

Next we recall our discretization scheme for this problem as given in [15]. We note setting $x(t) = t$ and $y(t) = f(t)$ is the same as saying $y = f(x)$. So we take x as the independent variable and $y(t) = y(x)$ as the dependent variable for this problem. We follow [15], and discretize the physical space consisting of the rectangle $[0, \pi] \times [0, 2]$ in $\mathbb{R}^2$. We let $x_k = \frac{k\pi}{\zeta}$ for $k = 0, 1 \cdots \zeta$. We set $y_0 = 2$ and $y_\zeta = 0$ to take into account the boundary condition. Let $y_k = y(x_k) \in \{y_{[i]} = \frac{2i}{L} : i = 0, 1 \cdots L\}$ for $k = 1$ to $\zeta - 1$. Next one can use Lagrange interpolation or Spline interpolation to determine the function $y(x)$. For example, on using Lagrange interpolation we can set $y(x) = \sum_{k=0}^{\zeta} y_k \phi_k(x)$. Here $\phi_k(x)$ is chosen such that $\phi_k(x_j) = 1$ if $j = k$ and $\phi_k(x_j) = 0$ otherwise. More explicitly

$$\phi_k(x) = \prod_{j=0,\ j\neq k}^{\zeta} \frac{(x - x_j)}{x_k - x_j}. \tag{6.2}$$

Essentially, it means that once a discretization of x is carried out, then we impose the condition that at $x = x_k$, the corresponding $y(x_k)$ can only have values from the set $\{y_{[i]} = \frac{2i}{L} : i = 0, 1 \cdots L\}$ (see Fig. 1). We note that with the above discretization there are $N = (L + 1)^{\zeta - 1}$ total possible cases need to be considered.

Now we follow the following steps to obtain a solution to the brachistochrone problem described earlier.

Step I. As $X_1(t) = x(t)$ is treated as an independent variable, we only need to consider $X_2(t) = y(t)$. Let $\tilde{X}_2 = [y_1, y_2, \cdots y_{\zeta-1}]$ be the random vector corresponding to $X_2(t) = y(t)$. We set $\zeta = 15$ and $L = 15$. We note that $y_0 = 2$ and $y_\zeta = 0$ because of boundary conditions.

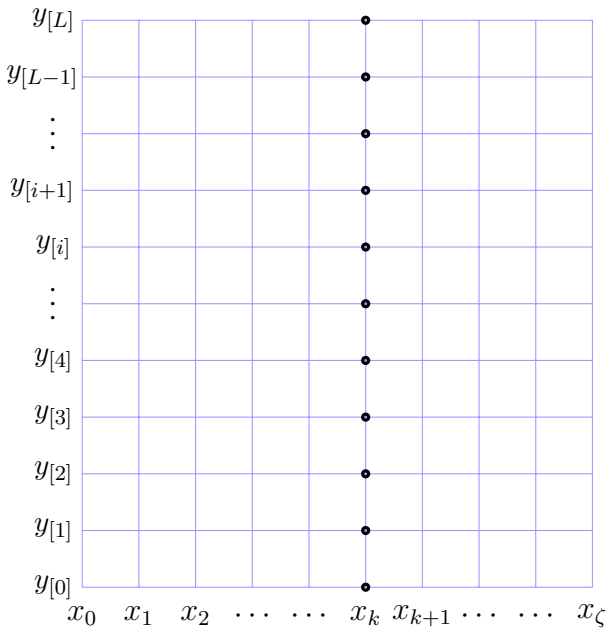

Figure 1: Brachistochrone problem: discretization of both the dependent variable y as well as the independent variable x is carried out. Note that at $x = x_k$, the corresponding $y_k = y(x_k)$ is allowed to take values only from the set $\{y_{[i]} : i = 0, 1 \cdots L\}$. The possible values for $y(x_k)$ is shown with thick black dots.

Step II. We carry out the random search algorithm (see Sect. 4.2, [15]) for $M = 30000$ number of times using the random values for the vector $[y_1, y_2, y_3, y_4, \cdots y_{14}]$. Here we note that one can use a smaller value of ζ, say $\zeta = 6$ to perform the random search algorithm, and thereafter using Lagrange interpolation the function $y(t)$ can be determined. Once $y(x)$ is known a more finer discretization can be picked, such as the one used earlier with $\zeta = 15$.

Step III. We pick the top $R = 50$ best solutions. Using these values and Lagrange or Spline interpolation we can get the trajectory $y(x)$ for each of these 50 rows. In other words, we now have 50 best solutions. Moreover using the function $y(x)$ obtained for each row we can obtain 50 sample values of the random vector $\tilde{X}_2$ (as shown below).

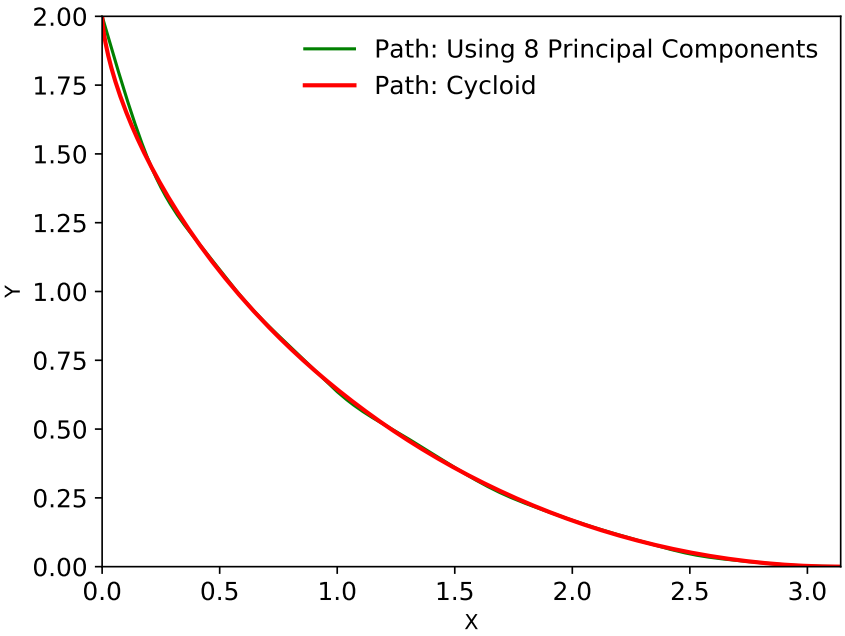

Figure 2: The optimum trajectory obtained for brachistochrone problem by a PCA based approach using 8 principal components. The correct analytical trajectory is plotted in red.

$$
\begin{pmatrix}
1.5 & 1.0 & 0.75 & 0.62 & 0.50 & 0.38 & 0.31 & 0.25 & 0.25 & 0.25 & 0.19 & 0.047 & -0.027 & -0.031 \\
1.8 & 1.5 & 1.2 & 1.2 & 1.0 & 1.0 & 1.0 & 0.88 & 0.75 & 0.62 & 0.62 & 0.62 & 0.44 & 0.25 \\
1.2 & 0.88 & 0.50 & 0.44 & 0.38 & 0.25 & 0.055 & -0.020 & 0.027 & 0.12 & 0.22 & 0.25 & 0.25 & 0.12 \\
1.5 & 1.0 & 0.75 & 0.62 & 0.50 & 0.44 & 0.31 & 0.25 & 0.31 & 0.38 & 0.38 & 0.31 & 0.22 & 0.11 \\
1.8 & 1.2 & 1.0 & 0.75 & 0.50 & 0.38 & 0.25 & 0.31 & 0.44 & 0.50 & 0.50 & 0.44 & 0.31 & 0.16 \\
1.8 & 1.5 & 1.5 & 1.2 & 1.0 & 0.88 & 0.75 & 0.62 & 0.50 & 0.50 & 0.50 & 0.62 & 0.50 & 0.25 \\
1.2 & 0.75 & 0.38 & 0.31 & 0.25 & 0.19 & 0.12 & 0.16 & 0.25 & 0.38 & 0.44 & 0.44 & 0.31 & 0.19 \\
1.5 & 1.0 & 0.62 & 0.44 & 0.25 & 0.16 & 0.12 & 0.16 & 0.25 & 0.38 & 0.50 & 0.50 & 0.50 & 0.25 \\
1.8 & 1.2 & 1.0 & 0.88 & 0.62 & 0.50 & 0.31 & 0.22 & 0.16 & 0.12 & 0.12 & 0.12 & 0.12 & 0.062 \\
1.8 & 1.5 & 1.2 & 0.88 & 0.62 & 0.44 & 0.22 & 0.094 & 0.078 & 0.12 & 0.22 & 0.25 & 0.25 & 0.16 \\
1.5 & 1.2 & 1.2 & 1.0 & 1.0 & 0.88 & 0.62 & 0.44 & 0.44 & 0.38 & 0.38 & 0.31 & 0.22 & 0.11 \\
1.2 & 0.62 & 0.31 & 0.38 & 0.50 & 0.50 & 0.44 & 0.31 & 0.22 & 0.12 & 0.12 & 0.12 & 0.12 & 0.078 \\
1.2 & 0.88 & 0.62 & 0.50 & 0.50 & 0.44 & 0.31 & 0.25 & 0.25 & 0.25 & 0.31 & 0.31 & 0.22 & 0.12 \\
1.2 & 0.88 & 0.50 & 0.50 & 0.50 & 0.62 & 0.62 & 0.62 & 0.62 & 0.50 & 0.38 & 0.22 & 0.078 & 0.023 \\
1.5 & 1.0 & 0.62 & 0.38 & 0.25 & 0.25 & 0.25 & 0.25 & 0.19 & 0.12 & 0.12 & 0.12 & 0.12 & 0.062 \\
1.5 & 1.2 & 1.2 & 1.2 & 1.2 & 1.2 & 1.2 & 1.0 & 0.75 & 0.50 & 0.31 & 0.19 & 0.094 & 0.031 \\
1.2 & 0.75 & 0.44 & 0.44 & 0.50 & 0.62 & 0.50 & 0.50 & 0.50 & 0.38 & 0.25 & 0.062 & -0.039 & -0.039 \\
1.5 & 1.0 & 0.88 & 0.62 & 0.50 & 0.31 & 0.078 & -0.027 & 0.020 & 0.12 & 0.22 & 0.25 & 0.25 & 0.12 \\
1.8 & 1.5 & 1.5 & 1.2 & 1.0 & 0.88 & 0.88 & 0.75 & 0.75 & 0.62 & 0.50 & 0.38 & 0.19 & 0.094 \\
1.5 & 1.0 & 0.75 & 0.62 & 0.50 & 0.44 & 0.31 & 0.25 & 0.25 & 0.25 & 0.31 & 0.31 & 0.22 & 0.12 \\
1.5 & 1.0 & 0.75 & 0.75 & 0.75 & 0.75 & 0.62 & 0.62 & 0.50 & 0.50 & 0.50 & 0.44 & 0.31 & 0.19 \\
1.5 & 1.0 & 0.62 & 0.50 & 0.38 & 0.31 & 0.25 & 0.31 & 0.38 & 0.38 & 0.38 & 0.31 & 0.22 & 0.11 \\
1.5 & 0.88 & 0.50 & 0.31 & 0.25 & 0.31 & 0.38 & 0.44 & 0.44 & 0.38 & 0.38 & 0.31 & 0.22 & 0.11 \\
1.8 & 1.5 & 1.2 & 0.88 & 0.62 & 0.44 & 0.31 & 0.25 & 0.38 & 0.38 & 0.31 & 0.19 & 0.078 & 0.027 \\
1.2 & 0.62 & 0.25 & 0.16 & 0.12 & 0.094 & 0.027 & -0.020 & -0.031 & 0.00 & 0.062 & 0.12 & 0.12 & 0.078 \\
1.5 & 0.88 & 0.75 & 0.88 & 0.88 & 1.0 & 0.88 & 0.75 & 0.62 & 0.50 & 0.50 & 0.44 & 0.31 & 0.19 \\
1.8 & 1.5 & 1.5 & 1.2 & 1.0 & 0.88 & 0.75 & 0.62 & 0.50 & 0.50 & 0.50 & 0.62 & 0.50 & 0.25 \\
1.5 & 1.2 & 1.2 & 1.2 & 1.0 & 0.88 & 0.50 & 0.25 & 0.078 & 0.00 & 0.031 & 0.11 & 0.12 & 0.094 \\
1.8 & 1.5 & 1.2 & 1.2 & 1.2 & 1.0 & 1.0 & 0.88 & 0.75 & 0.50 & 0.38 & 0.19 & 0.094 & 0.031 \\
1.5 & 1.0 & 0.75 & 0.50 & 0.38 & 0.31 & 0.25 & 0.25 & 0.25 & 0.25 & 0.31 & 0.38 & 0.38 & 0.22 \\
1.5 & 1.2 & 1.0 & 0.88 & 0.75 & 0.75 & 0.62 & 0.44 & 0.31 & 0.25 & 0.25 & 0.25 & 0.25 & 0.12 \\
1.2 & 0.50 & 0.094 & -0.031 & 0.00 & 0.11 & 0.22 & 0.31 & 0.31 & 0.25 & 0.16 & 0.039 & -0.023 & -0.027 \\
1.5 & 1.0 & 0.88 & 0.75 & 0.62 & 0.62 & 0.62 & 0.62 & 0.62 & 0.50 & 0.44 & 0.31 & 0.22 & 0.11 \\
1.2 & 0.88 & 0.62 & 0.50 & 0.50 & 0.50 & 0.44 & 0.38 & 0.38 & 0.38 & 0.50 & 0.50 & 0.50 & 0.25 \\
1.5 & 1.0 & 0.75 & 0.50 & 0.25 & 0.16 & 0.12 & 0.16 & 0.22 & 0.25 & 0.25 & 0.16 & 0.094 & 0.047 \\
1.5 & 1.2 & 0.88 & 0.62 & 0.38 & 0.22 & 0.12 & 0.16 & 0.25 & 0.38 & 0.44 & 0.44 & 0.31 & 0.19 \\
1.5 & 1.0 & 0.75 & 0.62 & 0.38 & 0.25 & 0.16 & 0.16 & 0.25 & 0.38 & 0.44 & 0.44 & 0.31 & 0.19 \\
1.5 & 1.2 & 1.0 & 0.88 & 0.75 & 0.75 & 0.62 & 0.50 & 0.50 & 0.50 & 0.50 & 0.44 & 0.31 & 0.19 \\
1.2 & 0.62 & 0.31 & 0.22 & 0.25 & 0.25 & 0.16 & 0.11 & 0.11 & 0.12 & 0.16 & 0.16 & 0.11 & 0.062 \\
1.5 & 1.2 & 1.0 & 0.88 & 0.88 & 0.88 & 0.88 & 0.75 & 0.75 & 0.62 & 0.75 & 0.75 & 0.62 & 0.31 \\
1.5 & 1.0 & 0.62 & 0.44 & 0.25 & 0.12 & 0.031 & -0.016 & -0.016 & 0.00 & 0.0059 & 0.0024 & -0.0012 & -0.0015 \\
1.5 & 1.0 & 0.75 & 0.62 & 0.62 & 0.75 & 0.75 & 0.75 & 0.75 & 0.62 & 0.50 & 0.38 & 0.19 & 0.094 \\
1.8 & 1.5 & 1.2 & 1.2 & 1.0 & 0.88 & 0.75 & 0.62 & 0.38 & 0.25 & 0.25 & 0.25 & 0.25 & 0.12 \\
1.5 & 0.88 & 0.50 & 0.38 & 0.25 & 0.19 & 0.12 & 0.16 & 0.25 & 0.38 & 0.44 & 0.44 & 0.31 & 0.19 \\
1.5 & 1.0 & 0.88 & 0.75 & 0.75 & 0.75 & 0.75 & 0.62 & 0.62 & 0.50 & 0.31 & 0.094 & -0.047 & -0.055 \\
1.5 & 1.2 & 1.0 & 0.88 & 0.75 & 0.75 & 0.62 & 0.62 & 0.62 & 0.62 & 0.62 & 0.50 & 0.31 & 0.16 \\
1.5 & 1.2 & 1.0 & 0.88 & 0.88 & 1.0 & 1.0 & 0.88 & 0.88 & 0.75 & 0.62 & 0.50 & 0.31 & 0.16 \\
1.8 & 1.5 & 1.2 & 1.2 & 1.0 & 0.88 & 0.75 & 0.62 & 0.62 & 0.50 & 0.44 & 0.31 & 0.22 & 0.094 \\
1.5 & 1.0 & 0.88 & 0.88 & 0.88 & 0.88 & 0.62 & 0.44 & 0.38 & 0.38 & 0.44 & 0.44 & 0.38 & 0.19 \\
1.8 & 1.2 & 1.0 & 0.88 & 0.62 & 0.50 & 0.31 & 0.22 & 0.19 & 0.12 & 0.078 & 0.020 & -0.012 & -0.014
\end{pmatrix}
$$

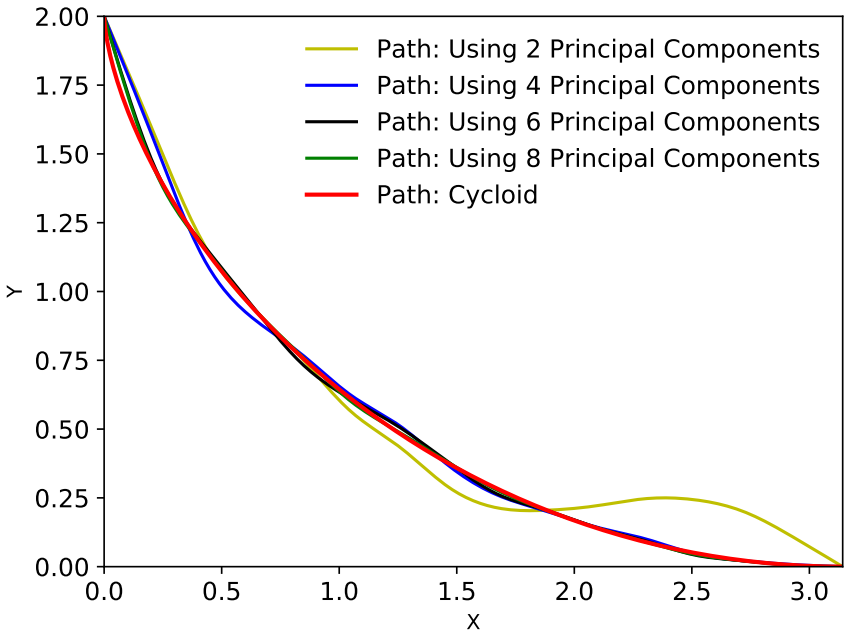

Figure 3: The comparison of optimum trajectory obtained for brachistochrone problem by a PCA based approach using different number of principal components. The correct analytical trajectory is plotted in red. It is clear that with more principal components the solution obtained is closer to the correct analytical solution.

Step IV. Next we perform PCA by considering $k = 8$ principal components. There are many software packages which support PCA. We used the python package scikit-learn (see [12]) for performing PCA. The covariance matrix that we obtained is given below for the purpose of illustration.

$$\begin{pmatrix}
0.027 & 0.039 & 0.047 & 0.039 & 0.027 & 0.020 & 0.020 & 0.014 & 0.0098 & 0.0078 & 0.0059 & 0.0059 & 0.0039 & 0.0020 \\
0.039 & 0.078 & 0.094 & 0.078 & 0.062 & 0.047 & 0.039 & 0.031 & 0.023 & 0.020 & 0.014 & 0.016 & 0.012 & 0.0059 \\
0.047 & 0.094 & 0.11 & 0.11 & 0.078 & 0.078 & 0.062 & 0.047 & 0.031 & 0.023 & 0.020 & 0.020 & 0.014 & 0.0068 \\
0.039 & 0.078 & 0.11 & 0.11 & 0.094 & 0.078 & 0.078 & 0.055 & 0.039 & 0.027 & 0.020 & 0.020 & 0.012 & 0.0059 \\
0.027 & 0.062 & 0.078 & 0.094 & 0.094 & 0.094 & 0.078 & 0.062 & 0.047 & 0.027 & 0.020 & 0.016 & 0.0098 & 0.0039 \\
0.020 & 0.047 & 0.078 & 0.078 & 0.094 & 0.094 & 0.094 & 0.078 & 0.055 & 0.031 & 0.023 & 0.016 & 0.0098 & 0.0039 \\
0.020 & 0.039 & 0.062 & 0.078 & 0.078 & 0.094 & 0.094 & 0.078 & 0.062 & 0.039 & 0.027 & 0.020 & 0.0078 & 0.0029 \\
0.014 & 0.031 & 0.047 & 0.055 & 0.062 & 0.078 & 0.078 & 0.078 & 0.062 & 0.039 & 0.027 & 0.020 & 0.0078 & 0.0029 \\
0.0098 & 0.023 & 0.031 & 0.039 & 0.047 & 0.055 & 0.062 & 0.062 & 0.055 & 0.039 & 0.031 & 0.020 & 0.0078 & 0.0029 \\
0.0078 & 0.020 & 0.023 & 0.027 & 0.027 & 0.031 & 0.039 & 0.039 & 0.039 & 0.031 & 0.027 & 0.023 & 0.012 & 0.0059 \\
0.0059 & 0.014 & 0.020 & 0.020 & 0.020 & 0.023 & 0.027 & 0.027 & 0.031 & 0.027 & 0.027 & 0.027 & 0.020 & 0.0098 \\
0.0059 & 0.016 & 0.020 & 0.020 & 0.016 & 0.016 & 0.020 & 0.020 & 0.020 & 0.023 & 0.027 & 0.031 & 0.027 & 0.014 \\
0.0039 & 0.012 & 0.014 & 0.012 & 0.0098 & 0.0098 & 0.0078 & 0.0078 & 0.0078 & 0.012 & 0.020 & 0.027 & 0.023 & 0.014 \\
0.0020 & 0.0059 & 0.0068 & 0.0059 & 0.0039 & 0.0039 & 0.0029 & 0.0029 & 0.0029 & 0.0059 & 0.0098 & 0.014 & 0.014 & 0.0078
\end{pmatrix}.$$

The eight principal components are given below, with each row representing a principal component. $\tilde{Z}_2 =$

$$\begin{pmatrix}
-0.12 & -0.25 & -0.38 & -0.38 & -0.38 & -0.38 & -0.38 & -0.31 & -0.22 & -0.16 & -0.12 & -0.094 & -0.062 & -0.027 \\
0.25 & 0.38 & 0.44 & 0.31 & 0.062 & -0.16 & -0.31 & -0.38 & -0.38 & -0.25 & -0.22 & -0.12 & -0.039 & -0.016 \\
-0.11 & -0.19 & -0.11 & 0.047 & 0.19 & 0.25 & 0.22 & 0.11 & -0.039 & -0.25 & -0.44 & -0.50 & -0.44 & -0.25 \\
-0.38 & -0.31 & -0.11 & 0.12 & 0.31 & 0.31 & 0.062 & -0.19 & -0.38 & -0.31 & -0.055 & 0.25 & 0.38 & 0.22 \\
-0.50 & -0.11 & 0.25 & 0.094 & 0.22 & 0.055 & -0.31 & -0.38 & 0.11 & 0.38 & 0.25 & -0.047 & -0.31 & -0.19 \\
0.62 & -0.44 & -0.31 & 0.31 & 0.22 & 0.078 & -0.22 & -0.16 & 0.012 & 0.16 & 0.16 & 0.023 & -0.11 & -0.027 \\
-0.12 & 0.62 & -0.62 & 0.16 & 0.22 & 0.020 & -0.094 & -0.078 & 0.016 & -0.023 & 0.16 & -0.078 & -0.027 & 0.0049 \\
0.31 & 0.16 & 0.078 & -0.75 & 0.22 & 0.44 & 0.0098 & -0.31 & 0.023 & 0.0024 & 0.11 & -0.055 & 0.020 & -0.039
\end{pmatrix}.$$

For example, from the above we see that the first principal component is

$$-0.12y_1 - 0.25y_2 - 0.38y_3 - 0.38y_4 - 0.38y_5 - 0.38y_6 - 0.38y_7 - 0.31y_8 - 0.22y_9$$

Table 1: A performance-comparison for the brachistochrone problem depending on the number of principal components used.

Number of principal components	Size of search space	Minimum time	Error %
2	100^2	1.03113613833	2.8
4	100^4	1.01720996497	1.4
6	100^6	1.00935407724	0.6
8	100^8	1.00816922477	0.5

$$-0.16y_{10} - 0.12y_{11} - 0.094y_{12} - 0.062y_{13} - 0.027y_{14}.$$

The explained variances of the principal components are given by:

$$0.69864 \quad 0.15927 \quad 0.089497 \quad 0.033484 \quad 0.007518 \quad 0.00438 \quad 0.002911 \quad 0.00174.$$

It means the first principal component explains 69.86% of the total variance in the data, the second principal component explains 15.93% of the total variance in the data, and so on.

Step V. Next we carry out discretization in the transformed space. As noted earlier a guiding factor in discretization in the transformed space could be the standard deviation in the direction of each of the principal component. We calculated the following standard deviation for the principal components.

$$0.7653 \quad 0.3654 \quad 0.2739 \quad 0.1675 \quad 0.0794 \quad 0.0606 \quad 0.0494 \quad 0.0382.$$

Each of the random variable can be restricted to take values in discrete set that lies within ± 3 standard deviations. For example, we can let the variable Z_{21} take values in the discrete set whose elements lie within the interval $(-3\sigma, 3\sigma)$, where $\sigma = 0.76528984$. In this step we may employ a finer discretization. For example, we can uniformly divide the interval $(-3\sigma, 3\sigma)$ into $L = 100$ smaller parts.

Step VI. Inverse transform is used to to get back to the original space.

Step VII. Exhaustive or random search algorithm given in (see Sect. 4, [15]) could be used to obtain the optimum trajectory.

In the brachistochrone example problem that we considered above, by using 8 principal components and with $L = 100$ in the Step V, the size of the search space is reduced from 100^{14} to 100^8. The minimum time we obtained using the above steps is 1.00816922 sec, whereas the correct analytical minimum time is 1.00354496 sec. Therefore, our machine learning based solution using 8 principal components is within 0.5% of the correct analytical

solution. In Fig. 2, we plotted the optimum trajectory obtained using 8 principal components (in black), as well as the trajectory obtained from the correct analytical solution (in red).

We used 8 principal components in the above example. Of course, we can use fewer principal components. In Table 1, we give a performance-comparison of the solution obtained depending on the number of principal components used. It is clear that the error is reduced on using more principal components. The plot in Fig. 3 shows the trajectory obtained using $2, 4, 6$ and 8 principal components. It illustrates the convergence of the obtained trajectory towards the correct analytical trajectory (i.e., a cycloid) as the number of principal components considered increases. For comparison, we note that using the random search algorithm with the traditional discretization scheme (without using PCA), with $\zeta = 8$ and $L = 15$, in 10000 trials the minimum time obtained was 1.1315974220770217 sec. The error with respect to the correct analytical minimum time was 12.76%. The advantage of the PCA based approach is clear in this example.

6.2 Isoperimetric problem

The objective here is to find the maximum area enclosed between a curve of the given fixed length and a given fixed straight line L, such that the endpoints of the curve must lie on the straight line L. This is the well known isoperimetric problem and it could be solved by employing the principles of calculus of variations. We will solve this problem by employing our algorithm.

Let $r = r(\theta)$, for $\theta = \frac{\pi}{2}$ to $\theta = \pi$, be a representation of the curve in polar coordinates with the boundary condition $r(\frac{\pi}{2}) = 0$. Let the length of the curve be $\frac{\pi}{3}$, i.e.,

$$\int_{\pi/2}^{\pi} \sqrt{r^2 + \left(\frac{dr}{d\theta}\right)^2}\, d\theta = \frac{\pi}{3}. \tag{6.3}$$

Let the fixed straight line be the line $\theta = \pi$. The objective function to be maximized is the area A given by

$$A = \int_{\pi/2}^{\pi} \frac{1}{2} r^2\, d\theta. \tag{6.4}$$

Next we explain our discretization scheme for this problem. The variable θ is treated as the independent variable and the variable $r = r(\theta)$ is assumed to be the dependent variable. Following [15], the parameter space consisting of the rectangle $[\pi/2, \pi] \times [0, b]$ in $\mathbb{R}^2$ is discretized, with a suitably chosen $b \leq \frac{\pi}{3}$. We let $\theta_k = \frac{k\pi}{\zeta}$ for $k = 0, 1 \cdots \zeta$. Let $r_k = r(\theta_k) \in \{r_{[i]} = \frac{bi}{L} : \ i = 0, 1 \cdots L\}$ for $k = 1$ to $\zeta - 1$. We set $r(\pi/2) = 0$ to take into account the boundary condition. Next we use either Lagrange interpolation or spline interpolation to determine the function $r = r(\theta)$.

It means that once a discretization of θ is carried out, then conditions are imposed such that at $\theta = \theta_k$, the corresponding $r(\theta_k)$ can only have values from the set $\{r_{[i]} = \frac{bi}{L} : \ i = 0, 1 \cdots L\}$ (see Fig. 4). At this point the function $r = r(\theta)$ may not satisfy the condition on its length, given by Eq. (6.3), so it may need to scale it by multiplying a constant c, i.e. set $r = cr(\theta)$, with c chosen such that Eq. (6.3) is now satisfied. We note that with the above discretization, there are $N = (L + 1)^{\zeta}$ total possible cases need to be considered.

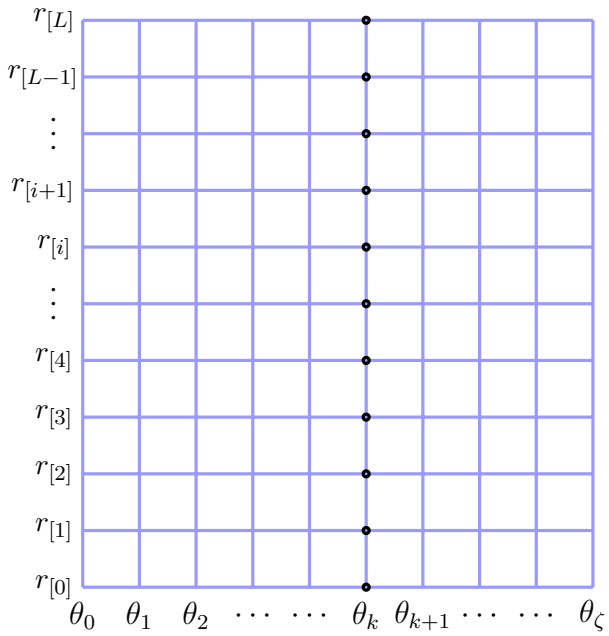

Figure 4: Isoperimetric problem : discretization of both the dependent variable r as well as the independent variable θ is carried out. Note that at $\theta = \theta_k$, the corresponding $r_k = r(\theta_k)$ is allowed to take values only from the set $\{r_{[i]} : i = 0, 1 \cdots L\}$. The possible values for $r(\theta_k)$ is shown with thick black dots.

Now we follow the following steps to obtain a solution to the isoperimetric problem described earlier.

Step I. As $X_1(t) = \theta(t)$ is treated as an independent variable, we only need to consider $X_2(t) = r(t)$. Let $\tilde{X}_2 = [r_1, r_2, \cdots r_\zeta]$ be the random vector corresponding to $X_2(t) = r(t)$. We set $b = 0.15$, $\zeta = 15$ and $L = 24$. We note that $r_0 = 0$ due to the boundary conditions.

Step II. We carry out the random search algorithm (see Sect. 4.2, [15]) for $M = 10000$ number of times using the random values for the vector $[r_1, r_2, r_3, \cdots r_{15}]$.

Step III. We pick the top $R = 50$ best solutions. We can get the trajectory $r(\theta)$ for each of these 50 rows by using Lagrange or spline interpolation. We now have 50 best solutions and by using the function $r(\theta)$ obtained for each row we can obtain 50 sample values of the random vector $\tilde{X}_2$.

The eight principal components are given below, with each row representing a principal component. $\tilde{Z}_2 =$

$$\begin{pmatrix}
-0.24 & -0.35 & -0.38 & -0.35 & -0.29 & -0.20 & -0.12 & -0.027 & 0.056 & 0.13 & 0.19 & 0.24 & 0.28 & 0.32 & 0.36 \\
0.35 & 0.40 & 0.28 & 0.11 & -0.059 & -0.19 & -0.28 & -0.30 & -0.27 & -0.19 & -0.067 & 0.076 & 0.22 & 0.34 & 0.37 \\
0.38 & 0.20 & -0.12 & -0.35 & -0.43 & -0.37 & -0.22 & -0.059 & 0.062 & 0.11 & 0.082 & 0.0024 & -0.10 & -0.24 & -0.45 \\
0.37 & 0.28 & 0.11 & 0.017 & 0.049 & 0.16 & 0.29 & 0.38 & 0.41 & 0.36 & 0.29 & 0.23 & 0.21 & 0.20 & 0.057 \\
0.19 & 0.070 & -0.093 & -0.17 & -0.15 & -0.036 & 0.092 & 0.17 & 0.16 & 0.032 & -0.18 & -0.38 & -0.45 & -0.19 & 0.65 \\
0.24 & -0.044 & -0.26 & -0.25 & -0.050 & 0.17 & 0.27 & 0.17 & -0.066 & -0.33 & -0.44 & -0.26 & 0.16 & 0.47 & -0.25
\end{pmatrix}$$

The explained variances of the principal components are given by:

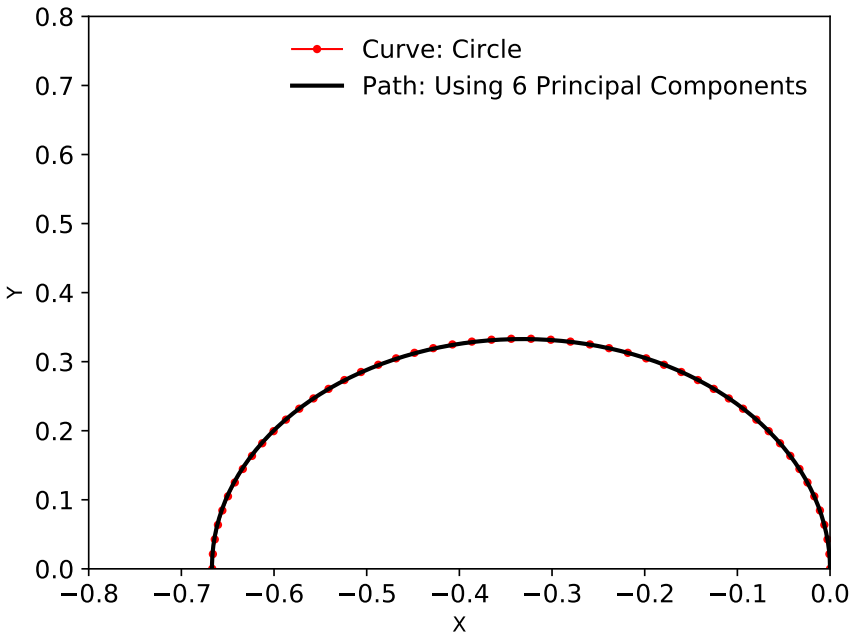

Figure 5: The optimum trajectory obtained for isoperimetric problem by a PCA based approach using 8 principal components. The correct analytical trajectory is plotted in red.

$$0.53040917 \quad 0.21786929 \quad 0.11771813 \quad 0.05851279 \quad 0.0481078 \quad 0.02738282.$$

It means the first principal component explains 53.04% of the total variance in the data, the second principal component explains 21.79% of the total variance in the data, and so on.

Step IV. Next discretization is carried out in the transformed space. We calculated the following standard deviation for the principal components.

$$0.23693071 \quad 0.15184967 \quad 0.11161884 \quad 0.07869394 \quad 0.07135487 \quad 0.05383379.$$

Similar to the previous example, we can let the variable Z_{21} take value in discrete set $(-3\sigma, 3\sigma)$, where $\sigma = 0.23693071$. In this step we may employ a finer discretization.

Step V. We use inverse transform to get back to the original space.

Step VI. We use the exhaustive or random search algorithm given in (see Sect. 4, [15]) to obtain the optimum trajectory.

The solution shown in Fig. 5 is obtained for the following values of the 6 principal components

$$0.3 \quad 0.08 \quad -0.13 \quad 0.19 \quad -0.02 \quad -0.04.$$

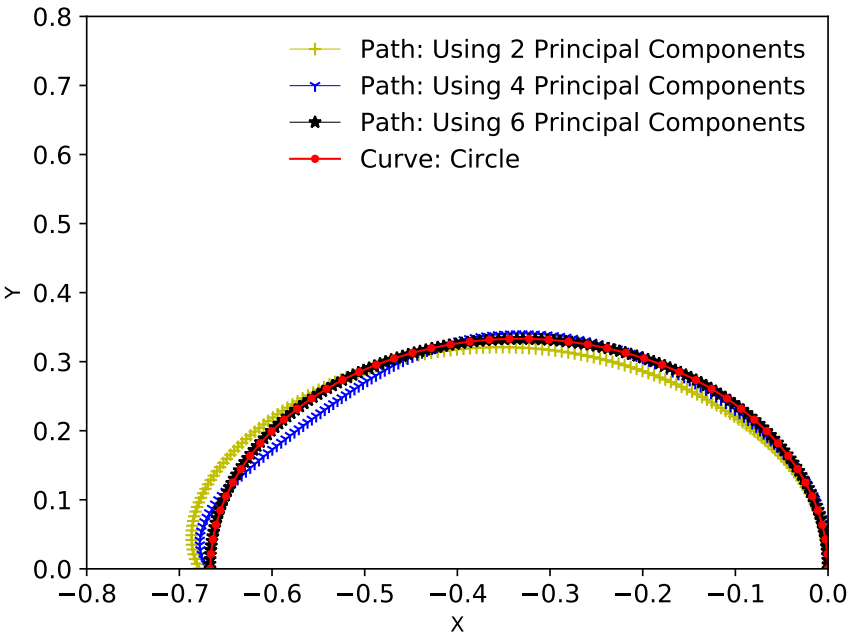

Figure 6: Comparison of optimum trajectory obtained for isoperimetric problem by a PCA based approach using different number of principal components. The correct analytical trajectory is plotted in red. It is clear that with more principal components the solution obtained is closer to the correct analytical solution.

Taking $L = 100$ in the Step V is sufficient to find the above solution for the principal components. Then, in this case, a very conservative estimate of the size of the search space is 100^6, whereas when a direct search is performed without using the principal components analysis, the size of the search space for a comparable solution would be 100^{15}. In fact, one can use lower values of L for less significant principal components such as the fourth, fifth and the sixth principal component in the above example, to have even a lower search space. The maximum area we obtained using the above steps is 0.174529529292298 sq. units, whereas the correct analytical area is $\frac{\pi}{18} \approx 0.17453292519$ sq. units. Therefore, our solution using 6 principal components is within 0.002% of the correct analytical solution.

In Fig. 5, we plotted the optimum trajectory obtained using 6 principal components (in black), as well as the trajectory obtained from the correct analytical solution (in red). We used 6 principal components in the above example. Of course, we can use fewer principal components. In Table 2, we give a performance-comparison of the solution obtained depending on the number of principal components used. The plot in Fig. 6 shows the trajectory obtained using $2, 4$ and 6 principal components.

6.3 Temperature control problem

As our next example we consider the problem of determining the control function that minimizes the energy required to heat a room. It is assumed that the objective here is to

Table 2: A performance-comparison for the isoperimetric problem depending on the number of principal components used.

Number of principal components	Size of search space	Minimum time	Error %
2	100^2	0.173450261720504	0.62
4	100^4	0.171702533474822	1.62
6	100^6	0.174529529292298	0.002

determine $u(t)$ for $t \in [0, 1]$, such that

$$J = \frac{1}{2} \int_0^1 u^2 \, dt, \tag{6.5}$$

under the following constraint and boundary conditions

$$\frac{dx}{dt} = -2x + u; \qquad x(0) = 0; \qquad x(1) = 10. \tag{6.6}$$

To solve this problem, we will discretize the space consisting of the rectangle $[0, 1] \times [1, 10]$ in $\mathbb{R}^2$. Let $t_k = \frac{k}{\zeta}$ for $k = 0, 1 \cdots \zeta$. Let $x_k = x(t_k) \in \{x_{[i]} = \frac{10\, i}{L} : \, i = 0, 1 \cdots L\}$ for $k = 1$ to $\zeta - 1$. Boundary conditions force $x_0 = 0$ and $x_\zeta = 10$. Next, similar to previous examples, Lagrange interpolation or Spline interpolation could be used to determine the function $x(t)$. Clearly, the above discretization results in a search space of size $N = (L + 1)^{\zeta - 1}$.

Step I. In this problem $X_1(t) = t$ is the independent variable and $X_2(t) = x(t)$ is the dependent variable. Let $\tilde{X}_2 = [x_1, x_2, \cdots x_{\zeta - 1}]$ be the random vector that corresponds to $X_2(t) = x(t)$. We set $\zeta = 8$ and $L = 15$. We note that $x_0 = 0$ and $x_\zeta = 10$ because of boundary conditions.

Step II. We carry out the random search algorithm (see Sect. 4.2, [15]) for $M = 20000$ number of times using the random values for the vector $[x_1, x_2, x_3, x_4, x_5, x_6, x_7]$.

Step III. We pick the top $R = 25$ best solutions. It means, now we have 25 sample values of the random vector $\tilde{X}_2$.

Step IV. Next we perform PCA by considering $k = 7$ principal components similar to the previous examples. We obtained the following covariance matrix.

$$\begin{pmatrix} 0.88 & 0.88 & 0.44 & 0.22 & 0.25 & 0.31 & 0.22 \\ 0.88 & 1.5 & 0.88 & 0.62 & 0.62 & 0.50 & 0.31 \\ 0.44 & 0.88 & 2.5 & 1.8 & 1.2 & 0.62 & 0.25 \\ 0.22 & 0.62 & 1.8 & 3.5 & 3.0 & 1.2 & 0.31 \\ 0.25 & 0.62 & 1.2 & 3.0 & 3.5 & 2.0 & 0.88 \\ 0.31 & 0.50 & 0.62 & 1.2 & 2.0 & 2.5 & 1.2 \\ 0.22 & 0.31 & 0.25 & 0.31 & 0.88 & 1.2 & 1.0 \end{pmatrix}.$$

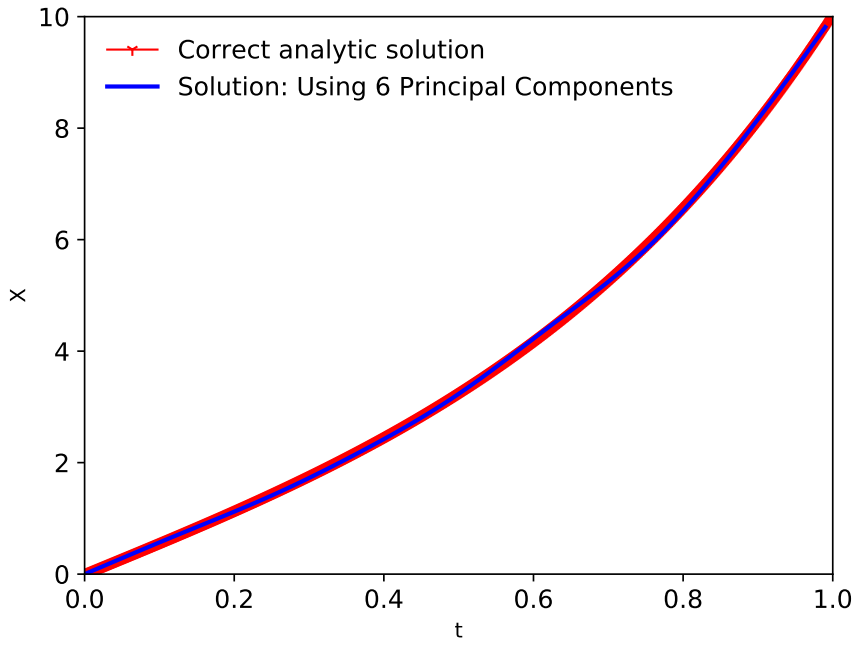

Figure 7: The optimum temperature function x(t) obtained for the temperature-control problem by a PCA based approach using 6 principal components. The correct analytical trajectory is plotted in red.

The seven principal components are given below, with each row representing a principal component. $\tilde{Z}_2 =$

$$\begin{pmatrix} 0.094 & 0.19 & 0.38 & 0.50 & 0.62 & 0.38 & 0.16 \\ -0.19 & -0.31 & -0.62 & -0.22 & 0.31 & 0.50 & 0.31 \\ 0.44 & 0.62 & 0.078 & -0.44 & -0.22 & 0.31 & 0.38 \\ -0.38 & -0.38 & 0.62 & -0.12 & -0.31 & 0.25 & 0.38 \\ 0.22 & -0.094 & -0.31 & 0.62 & -0.50 & 0.027 & 0.31 \\ 0.75 & -0.62 & 0.16 & -0.11 & 0.11 & -0.0098 & -0.078 \\ 0.0015 & -0.0059 & 0.020 & -0.078 & 0.31 & -0.62 & 0.62 \end{pmatrix}.$$

For example, from the above we see that the first principal component is

$$0.094x_1 + 0.19x_2 + 0.38x_3 + 0.50x_4 + 0.62x_5 + 0.38x_6 + 0.16x_7.$$

The explained variances of the principal components are given by:

$$0.585579287 \quad 0.161799859 \quad 0.138007909 \quad 0.0610509495$$
$$0.0342190627 \quad 0.0193429319 \quad 4.78637743 \times 10^{-32}.$$

It means the first principal component explains 58.56% of the total variance in the data, the second principal component explains 16.18% of the total variance in the data, and so on.

Step V. Next we carry out discretization in the transformed space. We note that a guiding factor in discretization in the transformed space could be the standard deviation in the direction of each of the principal component. We calculated the following standard deviation for the principal components.

$$2.92920710 \quad 1.53973611 \quad 1.42203149 \quad 0.945808985$$

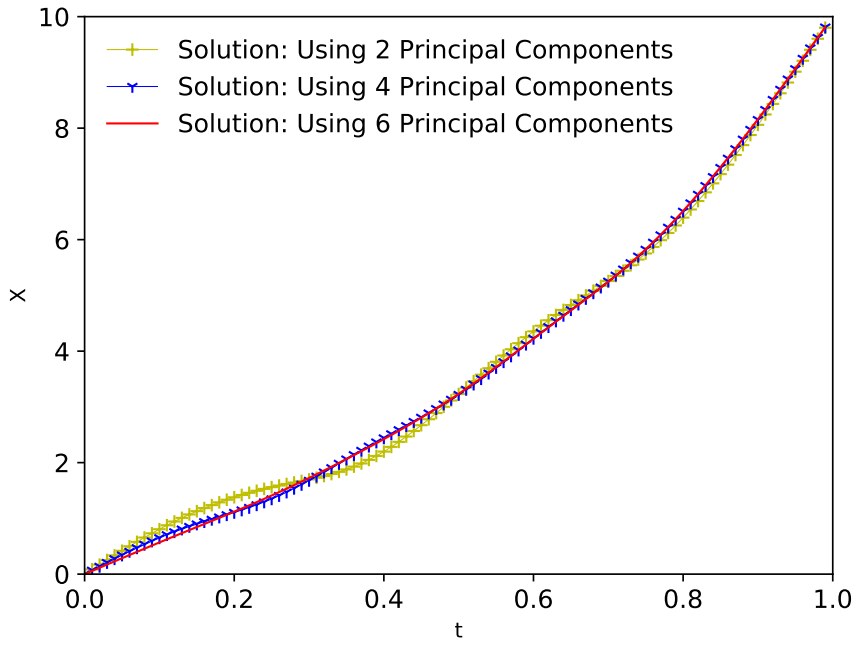

Figure 8: The comparison of temperature function x(t) obtained for the optimal temperature-control problem by a PCA based approach using different number of principal components.

$$0.708094594 \quad 0.532375885 \quad 4.57265638 \times 10^{-16}.$$

Each of the random variable can be restricted to take values in discrete set that lies within ± 3 standard deviation.

Step VI. We use inverse transform to get back to the original space.

Step VII. We use the exhaustive or random search algorithm given in (Sect. 4, [15]) to obtain the optimum solution.

The following values of the 6 principal components are used for the solution shown in Fig. 7.

$$0.7 \quad 0.5 \quad -0.1 \quad 0.4 \quad 0 \quad -0.1.$$

Table 3: A performance-comparison for the temperature-control problem depending on the number of principal components used.

Number of principal components	Size of search space	Minimum time	Error %
2	100^2	205.970746704	1.100
4	100^4	204.132527827	0.197
6	100^6	203.895971379	0.081

The minimum energy (cost), we obtained using the above steps by using 6 principal components, is 203.895971379 units, whereas the correct analytical solution is 203.73 units. Therefore, our machine learning based solution using 6 principal components is within 0.081% of the correct analytical solution. In Table 3, we give a performance-comparison of the solution obtained depending on the number of principal components used. The plot in Fig. 8 shows the trajectory obtained using $2, 4$ and 6 principal components. For comparison we note that, on using a direct traditional discretization (without PCA) scheme followed by the random search algorithm (Sect. 4, [15]), we obtained the value of minimum energy as 236.2631025205139 units in 20000 trials. The error with respect to the correct analytical solution is about 15.97%. The advantage of using a PCA based approach for discretization is very clear in this example problem.

Remark. It can be checked by using Euler-Lagrange equation (from calculus of variations), that for the temperature-control problem described above, the minimum J in Eq. (6.5), is obtained for

$$x(t) = \frac{10\,(e^{2t} - e^{-2t})}{e^2 - e^{-2}}. \tag{6.7}$$

The minimum value obtained is 203.73.

7 Alternate methods

We recall that a singular value decomposition (SVD) of an $m \times n$ matrix X can be expressed as

$$X = USV^T, \tag{7.1}$$

where U is an $m \times m$ orthogonal matrix, V is an $n \times n$ orthogonal matrix and S is an $m \times n$ diagonal matrix with nonnegative entries. The diagonal entries in S are called the singular values and they are arranged in the decreasing order of magnitude in S.

Now, we recall the well-known relation between PCA and SVD. Since, the covariance matrix X^TX is a symmetric matrix, we can write

$$X^TX = QDQ^T, \tag{7.2}$$

where Q is an $n \times n$ orthogonal matrix and D is an $n \times n$ diagonal matrix. The diagonal entries of D are the principal components. Now, from Eq. (7.1) and Eq. (7.2) we get

$$X^TX = (USV^T)^TUSV^T = V(S^TS)V^T, \tag{7.3}$$

and therefore, $D = S^TS$ and $Q = V$. It follows that computing PCA is reduced to computing SVD of a matrix. We note that the number of non-zero singular values is the rank of matrix X. There are a number of optimized and approximate methods available for computing SVD [9], [14].

We described a PCA based scheme to solve global trajectory optimization problems as well as problems related to calculus of variations. We note that although we ignored the cost

of performing PCA, in many cases when dealing with very high dimensional problems, this cost is significant and can not be ignored. In such cases it is reasonable to look for alternate methods. The low rank matrix approximation methods offer one such attractive method. Such methods are widely studied and investigated in literature [3], [1] and can be used to minimize computational costs.

8 Conclusion

We have shown that a PCA based method could be effectively utilized for solving a global trajectory optimization problem by exploiting the inherent structure present in the approximate solutions obtained by stochastic methods. In conjunction with our paper [15], the PCA based approach provides a complete framework for efficiently solving global trajectory optimization problems. PCA based approach allows us to perform discretization in a lower dimensional transformed space. The main advantage of the method presented in this paper is that it greatly reduces the size of the search space without significantly compromising the accuracy of the solution. This means, our proposed PCA based approach will also work even for such a high dimensional problem, for which the size of the search space would become prohibitively high, if the traditional discretization scheme is used. Another attractive feature, of our approach presented here, is the possibility of applying quantum computational algorithms (as presented in [15]) to solve global optimization problems.

The application of PCA in the context of global optimization problems is novel and we have shown with three representative examples that our proposed PCA based discretization method can be effectively employed to solve such problems.

References

[1] Dimitris Achlioptas and Frank McSherry. Fast computation of low-rank matrix approximations. Journal of the ACM (JACM), 54(2):9, 2007.

[2] John T Betts. Survey of numerical methods for trajectory optimization. Journal of Guidance control and dynamics, 21(2):193–207, 1998.

[3] Tony F Chan and Per Christian Hansen. Some applications of the rank revealing qr factorization. SIAM Journal on Scientific and Statistical Computing, 13(3):727–741, 1992.

[4] Christopher Chatfield and Alexander J Collins. Principal component analysis. In Introduction to multivariate analysis, pages 57–81. Springer, 1980.

[5] Ian D Cowling, James F Whidborne, and Alastair K Cooke. Optimal trajectory planning and lqr control for a quadrotor uav. In UKACC International Conference on Control, 2006.

[6] Brian S Everitt, Graham Dunn, et al. Applied multivariate data analysis, volume 2. Wiley Online Library, 2001.

[7] Fariba Fahroo and I Michael Ross. Direct trajectory optimization by a chebyshev pseudospectral method. Journal of Guidance, Control, and Dynamics, 25(1):160–166, 2002.

[8] Lov K Grover. A fast quantum mechanical algorithm for database search. In Proceedings of the twenty-eighth annual ACM symposium on Theory of computing, pages 212–219. ACM, 1996.

[9] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. SIAM review, 53(2):217–288, 2011.

[10] R.A. Johnson and D.W. Wichern. Applied Multivariate Statistical Analysis. Applied Multivariate Statistical Analysis. Pearson Prentice Hall, 2007.

[11] Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. Journal of Global optimization, 13(4):455–492, 1998.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12:2825–2830, 2011.

[13] Qazi Ibadur Rahman, Gerhard Schmeisser, et al. Analytic Theory of Polynomials. Number 26. Oxford University Press, 2002.

[14] Vladimir Rokhlin, Arthur Szlam, and Mark Tygert. A randomized algorithm for principal component analysis. SIAM Journal on Matrix Analysis and Applications, 31(3):1100–1124, 2009.

[15] Alok Shukla and Prakash Vedula. Trajectory optimization using quantum computing. Journal of Global Optimization, 2019. DOI: https://doi.org/10.1007/s10898-019-00754-5.

[16] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In Advances in neural information processing systems, pages 2951–2959, 2012.

[17] Oskar Von Stryk and Roland Bulirsch. Direct and indirect methods for trajectory optimization. Annals of operations research, 37(1):357–373, 1992.

[18] Ziyu Wang, Masrour Zoghi, Frank Hutter, David Matheson, Nando De Freitas, et al. Bayesian optimization in high dimensions via random embeddings. In IJCAI, pages 1778–1784, 2013.

[19] Jian Wu and Peter Frazier. The parallel knowledge gradient method for batch bayesian optimization. In Advances in Neural Information Processing Systems, pages 3126–3134, 2016.