Solving trajectory optimization problems via nonlinear programming: the brachistochrone case study

Jean-Pierre Dussault

February 22, 2012

Abstract

This note discusses reformulations the brachistochrone problem suitable for solution via NLP. The availability of solvers and modeling languages such as AMPL [1] makes it tempting to formulate discretized optimization problems and get solutions to the discretized versions of trajectory optimization problems. We use the famous brachistochrone problem to warn that the resulting solutions may be far different from the true optimal trajectory. Actually, we use our knowledge of the brachistochrone to argue that without this knowledge, we could not distinguish the true solution (a cycloid) from spurious solutions obtained by a natural discretization.
Keywords: trajectory optimization — unconstrained optimization

1 Introduction

In [5], four simple case studies are analyzed from a pedagogical point of view, among which the well known brachistochrone problem. Many issues are discussed and several lessons are drawn from the cases. An important concern in this paper is the impact of various formulations on the behavior of a non linear program solver.

The author [5] reports much difficulty with the brachistochrone and eventually resorts to vertical displacements. This approach fails to solve the low slope case. Actually, as we will see, the cycloid is *not* an optimal solution for the mid point discretization[1]. The lesson to be drawn, discretized optimal trajectories for trajectory optimization problems may produce non optimal solutions for the discretized version of the optimization problem. To repeat differently: for the brachistochrone, the discretized optimal cycloid is not optimal for the discretized brachistochrone problem.

This is a big warning since in the case of a real life problem for which no analytical solution is known, comparison with the exact solution is out of question and careful analysis is clearly required to assess optimality.

This note examines this somewhat alarming phenomenon further. We will present briefly the classical problem and its analytical solution. Then, we will proceed to discretize the

[1]Terminology used in [5]

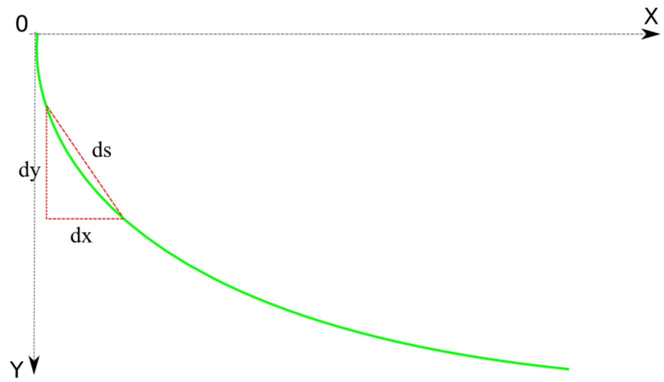

Figure 1: The situation

optimization problem to obtain an approximate solution using some NLP solver, and will conclude that the optimal solution thus reached does not correspond to the classical analytical solution. We will the provide an ad hoc improvement in order to obtain a discretized solution corresponding to the cycloid, the analytical solution. We will comment on the impact of the formulation on the NLP solver's behavior.

Last, we will discuss general procedures to avoid spurious discretized solutions. We will then examine the optimality conditions for the original trajectory problem and unfortunately conclude that without knowledge of the analytical solution, we could not discriminate between the spurious solution and the true cycloid solution.

2 The Brachistochrone problem

The famous toboggan problem consists in finding the shape of a track such that a ball launched on it reaches its end in minimal time. The shape will be parametrized by $s(\tau) = (x(\tau), y(\tau))$ in such a way that $s(0) = (0, 0)$ and $s(\tau^*) = (\hat{x}, \hat{y})$ with $\hat{x} \geq 0$ and $\hat{y} \leq 0$.

The optimization problem then reduces to

$$\min \int_0^{\tau^*} \frac{ds}{\sqrt{2gy}} \tag{1}$$

where g is the gravitational constant and $y \geq 0$ points downward. It is thus implicitly assumed that the optimal track remains below 0. Usually, taking advantage of the interpretation, one assumes that $x(\tau)$ is monotonic, express $ds = \sqrt{dx^2 + dy^2}$ to rewrite equation (1) as

$$\min \int_0^{\hat{x}} \sqrt{\frac{1 + \left(\frac{dy}{dx}\right)^2}{2gy}} dx \tag{2}$$

This is an instance of the general formulation

$$\min \int_0^{\hat{x}} L(x, y, y')dx. \tag{3}$$

For a curve $y^*(x)$ to be optimal, it is necessary that it satisfies the Euler-Lagrange equations:

$$\frac{\partial L}{\partial y} - \frac{d}{dt}\left(\frac{\partial L}{\partial y'}\right) = 0. \tag{4}$$

In our case, L does not involve x explicitly and the simplified necessary condition (Beltrami) is:

$$L - y'\frac{\partial L}{\partial y'} = constant. \tag{5}$$

After simplification to

$$(1 + y'^2)y = constant, \tag{6}$$

the solution may be obtained analytically, and is given by an appropriate portion of a cycloid:

$$x(\tau) = C(\tau - \sin(\tau)) \tag{7}$$
$$y(\tau) = C(\cos(\tau) - 1) \tag{8}$$

for suitable values of C^* and τ^*. Notice that C^* and τ^* are not available analytically but require some parameter identification computation.

3 Discretization

The discretization process to solve a trajectory problem consists in approximating the solution (optimal trajectory) by a suitable finite dimensional approximation (piecewise linear for instance). The basic approach then reduces to

$$\min \int_0^{\hat{x}} \sqrt{\frac{1+\left(\frac{dy}{dx}\right)^2}{2gy}}dx = \sum_{i=0}^{N} \int_{x_i}^{x_{i+1}} \sqrt{\frac{1+\left(\frac{dy}{dx}\right)^2}{2gy}}dx. \tag{9}$$

Moreover, if on each segment $[x_i, x_{i+1}]$, the function $y(x)$ is linear, the approximate solution is then a polygonal function joining the vertices (x_i, y_i). Therefore, $\frac{dy}{dx}$ is constant, and say evaluates to dyx_i so that the i^{th} segment's equation is given by $y(x) = y_i + dyx_i(x - x_i)$. A common technique consists in replacing the inner integral

$$\int_{x_i}^{x_{i+1}} \sqrt{\frac{1+\left(\frac{dy}{dx}\right)^2}{2gy}}dx \tag{10}$$

by some integration rule, often the so-called midpoint rule:

$$\int_{x_i}^{x_{i+1}} \sqrt{\frac{1+\left(\frac{dy}{dx}\right)^2}{-2gy}}dx \sim \sqrt{\frac{1+dyx_i^2}{2g\frac{(y_i+y_{i+1})}{2}}}(x_{i+1} - x_i) \tag{11}$$

where y_i is the y coordinate of the piecewise linear i^{th} vertex. Since $dyx_i = \frac{y_{i+1}-y_i}{x_{i+1}-x_i} = \frac{dy_i}{dx_i}$, the latter may be rewritten as

$$\sqrt{\frac{dx_i^2 + dy_i^2}{g(y_i + y_{i+1})}} \tag{12}$$

Common wisdom suggests that as $N \to \infty$, the solution polygonal trajectory (x_i, y_i) should approach the aforementioned cycloid. Unfortunately, it isn't so. Here follows the AMPL [1] code of this discretization:

```
param N := 32;
param eps := 0.0000000001;
param x{j in 0..N}:= (hatx)*(j/N);
var y {j in 0..N} ;

param dx {j in 1..N} := (x[j] - x[j-1]);
var dy {j in 1..N} = (y[j] - y[j-1]);
var s{j in 1..N} =  sqrt(dx[j]^2 + dy[j]^2);

var f {j in 1..N} = sqrt( (dx[j]^2 + dy[j]^2)/((y[j-1]+y[j])) );

minimize time: sum {j in 1..N} f[j] ;

subject to y0: y[0] = 0;
subject to yn: y[N] = 1;
```

The solution is illustrated in Figure 2. As it happens, the cycloid has a higher objective function value than the spurious discretized solution, and moreover, is not a stationary point for this objective function. The discretized problem is different from the original trajectory optimization problem. The figure remains qualitatively the same no matter how many discretization points are used.

4 An ad hoc correction

We present here an *ad hoc* correction to the naive discretization in order to approach the cycloid. For the piecewise linear trajectory, $\frac{dy}{dx}$ is constant on a piece, so that one has

$$\int_{x_i}^{x_{i+1}} \sqrt{\frac{1 + \left(\frac{dy}{dx}\right)^2}{2gy}} dx = \int_{x_i}^{x_{i+1}} \sqrt{\frac{1 + dyx_i^2}{2gy}} dx, \tag{13}$$

and the only quantity under the integral is y as a function of x. Therefore,

$$\int_{x_i}^{x_{i+1}} \sqrt{\frac{1 + dyx_i^2}{2gy}} dx = \sqrt{\frac{1 + dyx_i^2}{2g}} \int_{x_i}^{x_{i+1}} \frac{1}{\sqrt{y}} dx \tag{14}$$

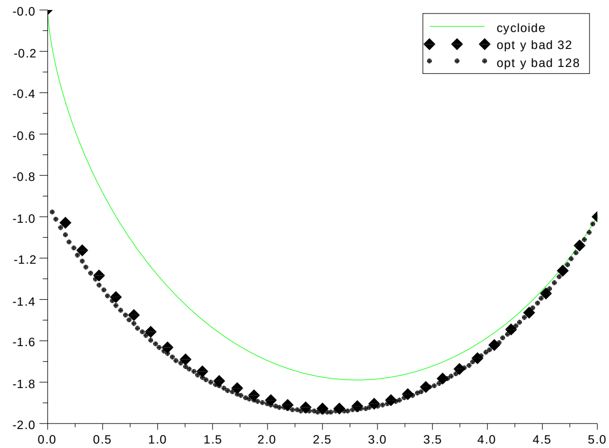

Figure 2: Spurious solution

and since the approximative trajectory is piecewise linear,

$$\int_{x_i}^{x_{i+1}} \frac{1}{\sqrt{y}} dx = \int_{x_i}^{x_{i+1}} \frac{1}{\sqrt{y_i + dyx_i(x - x_i)}} dx = 2\frac{x_{i+1} - x_i}{\sqrt{y_i} + \sqrt{y_{i+1}}} \tag{15}$$

which finally yields the simplified formulation

$$\int_{x_i}^{x_{i+1}} \sqrt{\frac{1 + \left(\frac{dy}{dx}\right)^2}{2gy}} dx = \frac{\sqrt{dx_i^2 + dy_i^2}}{\sqrt{gy_i} + \sqrt{gy_{i+1}}}. \tag{16}$$

This formulation allows to approximate the cycloid.

4.1 Independent x and y

We took for granted that y is a function of x. For a general problem, one should optimize the trajectory with respect to both x and y since one does not know in advance the property of the solution. This is easily done in AMPL by declaring x as “var” instead of “param”. Here follows the code:

```
param n := 32;
param eps := 0.0000000001;
param g   :=  9.81;
param xn := 5;  #  final x to reach
                #  final y == (-)1

var x {j in 0..n}:= xn*(j/n);
```

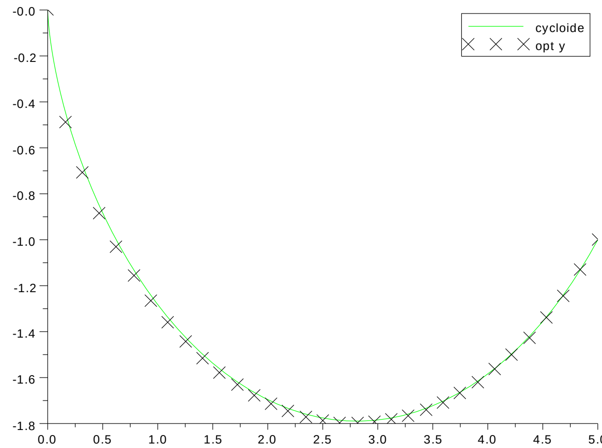

Figure 3: Good approximation to the cycloid

```
var y {j in 0..n}:= j/n ;

var dx {j in 1..n} = (x[j] - x[j-1]);
var dy {j in 1..n} = (y[j] - y[j-1]);

var s{j in 1..n} =  sqrt(dx[j]^2 + dy[j]^2);

var f {j in 1..n} = s[j] /(sqrt(max(g*y[j-1],eps))+sqrt(max(g*y[j],eps)));

minimize time: sum {j in 1..n} f[j] ;

subject to x0: x[0] = 0;
subject to x1: x[n] = xn;

subject to y0: y[0] = 0;
subject to y1: y[n] = 1;
```

Observe in figure 4 that sample points are closer near the origin, where “it counts”.

4.1.1 NLP solver behavior

In [5], the accent was put on the behavior of the NLP solver used (LOQO [4]). The main concern of the present paper is to obtain and assess the true cycloid solution. We nevertheless offer some comments on the challenges the formulations present to the solvers.

Actually, the NLP discretizations are all unconstrained problems. AMPL succeeds in getting rid of all the apparent constraints by substitutions. Therefore, any unconstrained

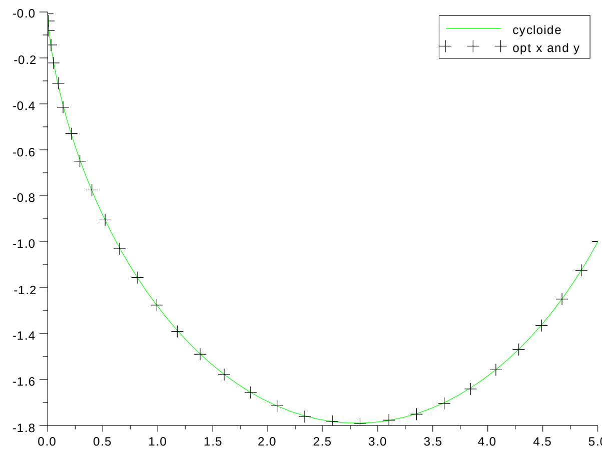

Figure 4: Better approximation to the cycloid

NLP solver may be used.

The ad hoc corrected version optimizing for fixed x (thus optimizing only in y) is an easy model. All the solvers we tried successfully got 10^{-8} accuracy on this model using more or less 100 function and gradient evaluations, even a naive version of steepest descent succeeded, albeit using more than 2000 evaluations.

In contrast, the ad hoc corrected optimizing both in x and y proves a challenge. When run at the "TryAMPL" site, many solvers reported failures (tn, L-BFGS-B, Minos, LOQO, Lancelot). We used mainly the optim command in Scilab, which is based on a limited memory BFGS algorithm developed by Gilbert and Lemaréchal [2] with success on all the model variants; optimizing in y only required 67 functions and gradients while optimizing in both x and y required 582 functions and gradients. Our in house truncated Newton failed (line search version) or was inefficient (trust region version).

5 General approach

Optimization of infinite dimensional functionals should best be done in the infinite dimension space, using appropriate approximations to estimate the function values and/or gradients. Therefore, the discretization of the solution (often a piecewise linear path) should not be imposed as a discretization of the objective functional.

Let us keep the piecewise linear form of the trajectory. Recall from equation (9) that the objective function is a then sum of pieces of the form

$$\int_{\tau_i}^{\tau_{i+1}} L(x, y, \dot{x}, \dot{y}) d\tau \qquad (17)$$

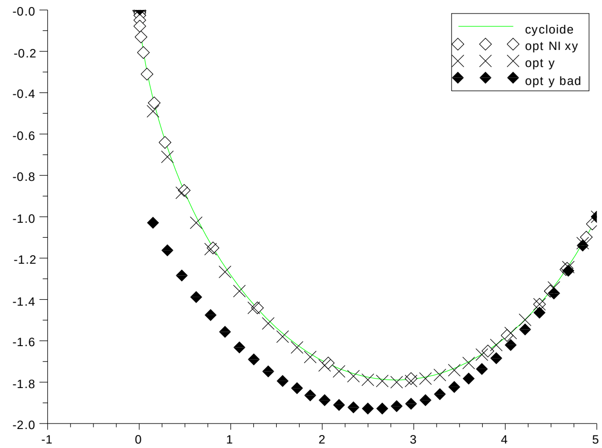

Figure 5: All solutions

which, since $x(\tau)$ and $y(\tau)$ are affine, may be rewritten as

$$\int_{\tau_i}^{\tau_{i+1}} L(x_i + \frac{\tau - \tau_i}{\tau_{i+1} - \tau_i}(x_{i+1} - x_i), y + \frac{\tau - \tau_i}{\tau_{i+1} - \tau_i}(y_{i+1} - y_i), dx_i, dy_i) d\tau \tag{18}$$

Therefore, for non trivial functions L, an explicit formula is not available to evaluate those integrals, and one needs to resort to numerical integration. One possibility is to use a simple Riemann sum:

$$\int_{\tau_i}^{\tau_{i+1}} L(x, y, \dot{x}, \dot{y}) d\tau \approx \frac{\tau_{i+1} - \tau_i}{M+1} \sum_{j=0}^{M} L(x_i + \frac{j}{M}(x_{i+1} - x_i), y_i + \frac{j}{M}(y_{i+1} - y_i), dx_i, dy_i) \tag{19}$$

The simplest case $M = 1$ corresponds to the so-called trapezoidal rule.

For example, for the brachistochrone problem, $M = 1$ is sufficient for converging to the cycloid. For $N = 32$ the solution of the ad hoc formulation (16) yields an objective function of 30.155218, and the approximation with $M = 1$ yields 30.415112 and with $M = 100$, one gets 30.161822. The figure 5 compares several solutions discussed in this paper, the optNI xy refering to the numerical integration using $M = 1$.

NI	objective
1	30.415112
10	30.188733
100	30.161822
∞	30.155218

6 Optimality conditions

In order to assess the quality of the resulting trajectory, it should be injected in the necessary condition for the original functional problem. In our case, the condition (6) and (5) should yield a constant vector. For $N = 32$, the table exhibits variances of the trajectory vectors. y' is given by $\frac{dy_i}{dx_i}$ while y is taken simply as $\frac{1}{2}(y_i + y_{i-1})$. Using the simplified condition (6) indeed exploit the knowledge of the problem and must be considered an *ad hoc* solution. We build the vector $V6_i = (1 + dydx_i^2)y_i$. It yields the following measures.

Trajectory	variance(V6)	$\frac{\max(V6)-\min(V6)}{\max(V6)}$
True Cycloid true y'	1.099×10^{-30}	3.474×10^{-15}
True Cycloid difference y'	0.0016426	0.1115434
Ad hoc x	0.0213924	0.3153419
Ad hoc xy	0.0999349	0.4999995
NI y=10	0.1372183	0.9984035
NI xy=10	0.2202405	0.9988068
Bad	13.422206	0.9150064

The good news is that the variance allows to identify the spurious bad solution, but the relative amplitude is worse for the numerical integration solution than for the spurious solution. The bad news is that using the plain Beltrami condition (5) without precaution, it falls in the same trap as the naive discretization since one has to evaluate L at appropriate values of $y(x)$. Indeed, the Beltrami equations simplify to

$$\sqrt{\frac{1+y'^2}{y}} - \frac{y'^2}{\sqrt{y(1+y'^2)}} = constant.$$

Again substitution $y'_i = \frac{dy_i}{dx_i}$ and $y \approx \frac{1}{2}(y_i + y_{i-1})$ in the above condition yields variances as follows:

Trajectory	variance(V5)	$\frac{\max(V6)-\min(V6)}{\max(V6)}$
True Cycloid true y'	2.326×10^{-30}	1.069×10^{-14}
True Cycloid difference y'	0.0000607	0.0574202
Ad hoc y	0.0005171	0.1721943
Ad hoc xy	0.0015167	0.2928929
NI y=10	10.501318	0.9600441
NI xy=10	12.425987	0.9654567
Bad	0.0081440	0.7084633

suggesting that the spurious solution more closely satisfies the necessary optimality conditions, more so than the NI approach.

We exhibit in Figure 6 the actual values of the optimality condition vectors. The spurious solution appears more constant than the NI solution; however, the value of the constant is

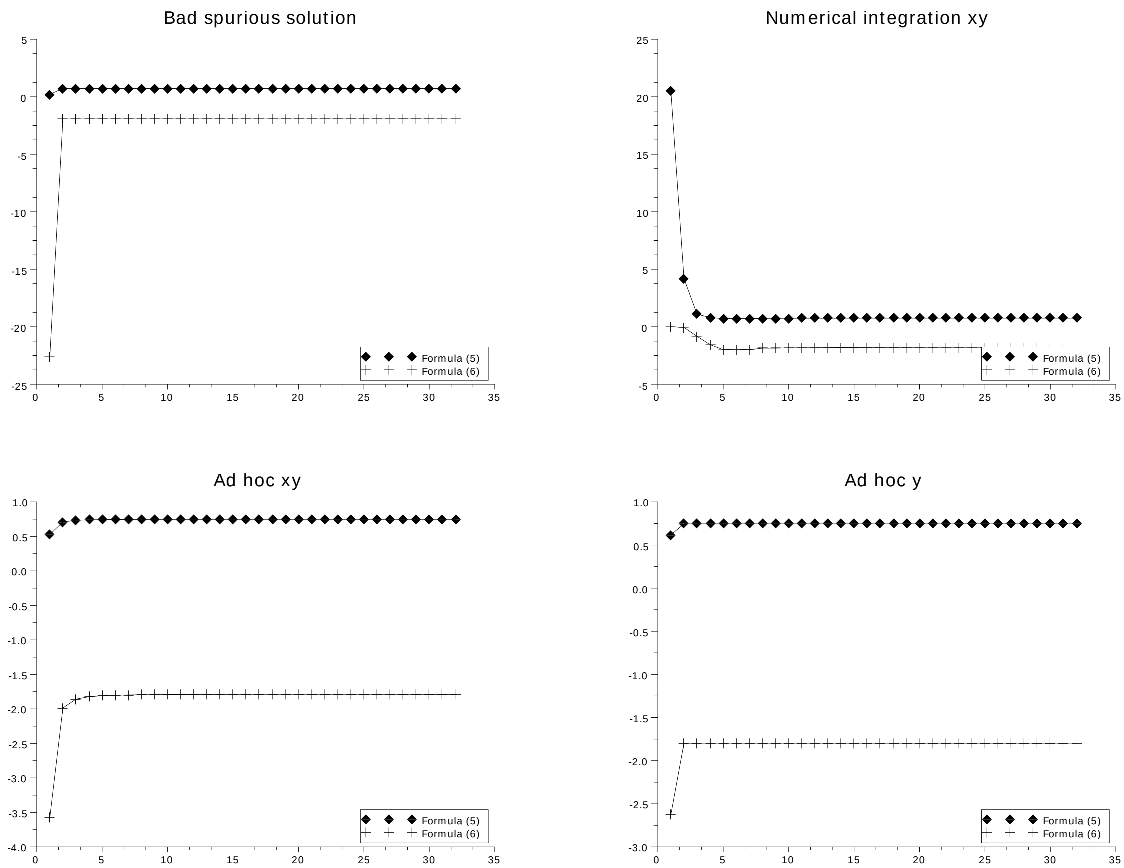

Figure 6: The optimality conditions

not the one corresponding to the optimal solution! Unfortunately, this is not apparent, and without knowledge of the true cycloid solution, perhaps one would arrive at the wrong conclusion that the spurious solution is the optimal one when compared to the NI one, despite the fact that the NI solution closely matches (visually) the cycloid.

Now, let us pretend we have to resort to the Euler-Lagrange conditions. Recall that $L(x, y, y') = \sqrt{\frac{1+y'^2}{y}}$.

$$\frac{\partial L}{\partial y} - \frac{d}{dx}\left(\frac{\partial L}{\partial y'}\right) = 0 \tag{20}$$

$$-\frac{\sqrt{1+y'^2}}{2\sqrt{y^3}} - \frac{d}{dx}\left(\frac{y'}{\sqrt{y(1+y'^2)}}\right) = 0 \tag{21}$$

$$-\frac{\sqrt{1+y'^2}}{2\sqrt{y^3}} - \frac{y''}{\sqrt{y(1+y'^2)}} + y'\frac{y'(1+y'^2)+2yy'y''}{2\sqrt{y^3(1+y'^2)^3}} = 0 \tag{22}$$

$$\frac{-(1+y'^2)^2 - 2yy''(1+y'^2) + y'^2(1+y'^2) + 2yy'^2y''}{2\sqrt{y^3(1+y'^2)^3}} \quad = \quad 0 \tag{23}$$

We observe that neglecting the denominator already constitutes an *ad hoc* simplification. To apply this condition to our piecewise linear path, we need to evaluate y''. Since we base our computation on mid points, we will use the mean value between naturally computed second derivatives at the end points of the intervals. Using the true path, $y' = \frac{-\sin(\tau)}{1-\cos(\tau)}$ and $y'' = \frac{1}{C(1-\cos(\tau)^2)}$, the norm of the residual to the Euler-Lagrange equation is 7.276×10^{-12}. Using finite difference approximations for y' and y'' but the true cycloid values for y yields 235.55359 so we did not pursue the study of those conditions, not even close to zero for the true trajectory.

7 Conclusion

Software tools that ease the formulation and solution of optimization problems are increasingly powerful. In [5], trajectory optimization case studies were addressed using AMPL [1]. We proposed a detailed study of the famous brachistochrone problem. While it is folklore that functional optimization problems such as trajectory optimization may possess different optimal solution than their discretized counterpart, it is remarkable that this classic example exhibits this phenomenon. Moreover, we present evidence that it is very difficult to distinguish the true solution from the spurious one. The use of optimality conditions may be misleading. I would trust that the increased accuracy in the evaluation of the discretized objective function yielding consistent solutions is an indicator to prefer the cycloid.

So, we arrive at a similar conclusion as Kahan [3] that mindless usage of powerful computational tools is indeed next to impossible and that nothing replaces careful mathematical analysis.

References

[1] R. Fourer, D.M. Gay, and B.W. Kernighan. *AMPL: a modeling language for mathematical programming*. Thomson/Brooks/Cole, 2003.

[2] Jean-Charles Gilbert and Claude Lemaréchal. Some numerical experiments with variable-storage quasi-newton algorithms. *Mathematical Programming*, 45:407–435, 1989.

[3] William M. Kahan. How futile are mindless assessments of roundoff in floating-point computation? `//www.cs.berkeley.edu/~wkahan/Mindless.pdf`.

[4] R. J. Vanderbei. LOQO user's manual - version 3.10. *Optimization Methods and Software*, 11:485–514, 1999.

[5] Robert Vanderbei. Case studies in trajectory optimization: Trains, planes, and other pastimes. *Optimization and Engineering*, 2:215–243, 2001. 10.1023/A:1013145328012.