ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, Illinois 60439

# COPS: Large-Scale Nonlinearly Constrained Optimization Problems

**Alexander S. Bondarenko, David M. Bortz, and Jorge J. Moré**

Mathematics and Computer Science Division

Technical Report ANL/MCS-TM-237

September 1998
October 1999 (Revision)

This work was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Computational and Technology Research, U.S. Department of Energy, under Contract W-31-109-Eng-38.

Contents

# COPS: Large-Scale Nonlinearly Constrained Optimization Problems

**Alexander S. Bondarenko*, David M. Bortz†, and Jorge J. Moré‡**

**Abstract**

We have started the development of COPS, a collection of large-scale nonlinearly **C**onstrained **O**ptimization **P**roblem**S**. The primary purpose of this collection is to provide difficult test cases for optimization software. Problems in the current version of the collection come from fluid dynamics, population dynamics, optimal design, and optimal control. For each problem we provide a short description of the problem, notes on the formulation of the problem, and results of computational experiments with general optimization solvers. We currently have results for DONLP2, LANCELOT, MINOS, SNOPT, and LOQO.

## 1 Introduction

COPS is a collection of large-scale nonlinearly **C**onstrained **O**ptimization **P**roblem**S**. We drew these test problems from a variety of sources, including some of the existing collections, such as the AMPL problems of R. Vanderbei,

`http://www.sor.princeton.edu/~rvdb/ampl/nlmodels/`,

the NETLIB collection of AMPL problems maintained by D. Gay,

`http://www.netlib.org/ampl/models/`,

the optimal control problems of Betts, Eldersveld, Huffman [4], and the MINPACK-2 collection [3]. We chose problems that arise in applications (for example, fluid dynamics, optimal shape design, population dynamics) or that have interesting features.

The aim of COPS is to challenge and test nonlinear optimization software. Users should note that this report describes work in progress. We expect that COPS will evolve and change as new problems appear and other researchers experiment with this collection. We welcome comments and suggestions for future directions.

We provide AMPL and C implementations. The problems in COPS are formulated as general constrained optimization problems defined by a merit function $f : \mathbb{R}^n \mapsto \mathbb{R}$ and nonlinear constraints $c : \mathbb{R}^n \mapsto \mathbb{R}^m$,

$$\min \{f(x) : x_l \le x \le x_u,\ c_l \le c(x) \le c_u\},$$

This work was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Computational and Technology Research, U.S. Department of Energy, under Contract W-31-109-Eng-38.

*Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213 (`asb@andrew.cmu.ed`).

†Department of Mathematics, North Carolina State University, Raleigh, North Carolina 27695 (`dmbortz@unity.ncsu.edu`).

‡Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Illinois 60439 (`more@mcs.anl.gov`).

where $x_l$ and $x_u$ are bounds on the variables, and $c_l$ and $c_u$ are bounds on the constraints.

The description of the problem as an optimization problem includes notes on the formulation and the structural information in Table 1.1. This information allows users to determine, in particular, the sparsity of the problem. We also include general comments on specific features and difficulties of the problems.

Table 1.1: Description of test problems

| |
|---|
| Variables |
| Constraints |
| Bounds |
| Linear equality constraints |
| Linear inequality constraints |
| Nonlinear equality constraints |
| Nonlinear inequality constraints |
| Nonzeros in $\nabla^2 f(x)$ |
| Nonzeros in $c'(x)$ |

An important component of this report is the inclusion of computational experiments with several general solvers (DONLP2, LANCELOT, MINOS, SNOPT, and LOQO), and comments on their behavior. We are well aware that these results will soon become obsolete as new versions of these packages become available. However, we feel that these results do provide a reasonable snapshot of the state of optimization software as of September 1998.

Finally, we provide plots of the solution for each problem. This is important so that users can verify that they obtained the correct solution. We feel that in many cases this is more useful and interesting than providing some measure of optimality.

Section 14 describes our C implementations, including the data structures used for each of problem. Implementations in AMPL and in C, along with sample drivers that use the C implementation with SNOPT, are available for downloading from our web site,

`http://www.mcs.anl.gov/~more/cops`.

## 2 Largest Small Polygon (Gay [8])

Find the polygon of maximal area, among polygons with $n_v$ sides and diameter $d \leq 1$.

### Formulation

The merit function is

$$f(r,\theta) = -\tfrac{1}{2}\sum_{i=1}^{n_v-1} r_{i+1} r_i \sin(\theta_{i+1}-\theta_i), \qquad r_{n_v} = 0, \quad \theta_{n_v} = \pi,$$

and the constraints are

$$\begin{aligned} r_i^2 + r_j^2 - 2r_i r_j \cos(\theta_i - \theta_j) \leq 1, \qquad & 1 \leq i \leq n_v - 2, \quad i+1 \leq j \leq n_v - 1, \\ \theta_i \leq \theta_{i+1}, \qquad & 1 \leq i \leq n_v - 2, \\ \theta_i \in [0,\pi],\ r_i \in [0,1], \qquad & 1 \leq i \leq n_v - 1. \end{aligned}$$

The optimal solution is not usually a regular hexagon, as was shown by Graham [9]. Another interesting feature of this problem is the presence of $O(n_v^2)$ nonlinear nonconvex inequality constraints and nonlinear nonconvex objective. We also note that as $n_v \to \infty$, we expect the maximal area to converge to the area of a unit-diameter circle, $\pi/4 \approx 0.7854$. This problem has many local minima. For example, for $n_v = 4$ a square with sides of length $1/\sqrt{2}$ and an equilateral triangle with another vertex added at distance 1 away from a fixed vertex are both global solutions with optimal value $f = \frac{1}{2}$. Indeed, the number of local minima is at least $O(n_v!)$. Thus, general solvers are usually expected to find only local solutions. Data for this problem appears in Table 2.1.

Table 2.1: Largest-small polygon problem data

| | |
|---|---|
| Variables | $n = 2(n_v - 1)$ |
| Constraints | $\frac{1}{8}n^2 + \frac{1}{4}n - 1$ |
| Bounds | $n$ |
| Linear equality constraints | 0 |
| Linear inequality constraints | $\frac{1}{2}n - 1$ |
| Nonlinear equality constraints | 0 |
| Nonlinear inequality constraints | $\frac{1}{8}n^2 - \frac{1}{4}n$ |
| Nonzeros in $\nabla^2 f(x)$ | $\frac{11}{2}n - 8$ |
| Nonzeros in $c'(x)$ | $\frac{1}{2}n^2 - 2$ |

### Performance

We provide results with the AMPL formulation on an SGI Onyx-2 Reality Monster. Results are summarized in Table 2.2. A polygon with almost equal sides was chosen as the standard starting guess for this problem. Global solutions for several $n_v$ are shown in Figure 2.1.

LANCELOT and SNOPT were successful at finding solutions for all $n_v$ tried. We also believe that these solutions are global solutions. SNOPT was more efficient than LANCELOT. MINOS was able to find only local solutions for $n_v \geq 15$.

Table 2.2: Performance of AMPL solvers

| Solver | $n_v = 6$ | $n_v = 10$ | $n_v = 20$ | $n_v = 50$ | $n_v = 100$ |
|---|---|---|---|---|---|
| DONLP2 | ‡ | ‡ | 1.5s | No * | No * |
| $f$ | 0.6749797629 | 0.7491366103 | 0.7768527183 | | |
| $\|c(x)\|$ | 1.23396E-06 | 2.61365E-07 | 5.8761E-07 | | |
| iterations | 12 | 24 | 38 | | |
| LANCELOT | ‡ | ‡ | 4s | 140s | 2899s |
| $f$ | 0.6749818114 | 0.7491373093 | 0.7768590578 | 0.7840156583 | 0.7850313647 |
| $\|c(x)\|$ | 7.3288E-06 | 3.6446E-06 | 6.5317E-06 | 2.1506E-06 | 3.8119E-06 |
| iterations | 16 | 18 | 50 | 116 | 228 |
| MINOS | ‡ | ‡ | 2s † | 45s † | 600s † |
| $f$ | 0.6749814429 | 0.7491373458 | 0.7687882291 | 0.734825561 | 0.7624733425 |
| $\|c(x)\|$ | 6.4E-13 | 2.1E-13 | 2.5E-13 | 8.5E-13 | 5.2E-11 |
| iterations | 30 | 49 | 497 | 1994 | 6948 |
| SNOPT | ‡ | ‡ | 0.2s | 8s | 861s |
| $f$ | 0.6749814429 | 0.7491373458 | 0.7768587560 | 0.7840161480 | 0.7850565708 |
| $\|c(x)\|$ | 2.0E-12 | 1.8E-11 | 8.3E-12 | 1.7E-09 | 1.4E-09 |
| iterations | 23 | 35 | 73 | 269 | 68152 |
| LOQO | ‡ | No | 53s † | No | No |
| $f$ | 0.6749814367 | failure | 0.7197409256 | failure | failure |
| dual $f$ | 0.6749814651 | | 0.7197409412 | | |
| iterations | 47 | 10000 | 537 | 10000 | 10000 |

† Local solution ‡ Global solution found in less than 0.1s * Problem is too large

LOQO was not able to solve the problem for most $n_v \geq 10$ that we tried with default parameters. However, we found that LOQO's performance improved slightly when setting the `mufactor` parameter small enough ($\approx 10^{-4}$), which is a scale factor for the barrier parameter [14, 15].

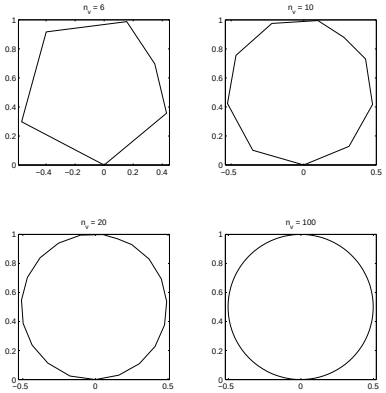


Figure 2.1: Unit-diameter polygons of maximal area

## 3 Distribution of Electrons on a Sphere (Vanderbei [13])

Given $n_p$ electrons, find the equilibrium state distribution (of minimal Coulomb potential) of the electrons positioned on a conducting sphere.

### Formulation

The merit function is

$$f(x,y,z) = \sum_{i=1}^{n_p-1} \sum_{j=i+1}^{n_p} \left( (x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2 \right)^{-\frac{1}{2}}$$

and the constraints are

$$x_i^2 + y_i^2 + z_i^2 = 1, \qquad i = 1, \ldots, n_p$$

Data for this problem appears in Table 3.1.

This problem, known as the Thomson problem of finding the lowest energy configuration of $n_p$ point charges on a conducting sphere, originated with Thomson's plum pudding model of the atomic nucleus. The Thomson problem is representative of an important class of problems in physics and chemistry of determining a structure with respect to atomic positions. This problem has many local minima at which the objective value is relatively close to the objective value at the global minimum. Also, the number of local minima grows exponentially [7, 10] with $n_p$. Thus, it is computationally difficult to determine the global minimum, and the solvers are usually expected to find only a local minimum.

Table 3.1: Electrons on a sphere problem data

| | |
|---|---|
| Variables | $n = 3n_p$ |
| Constraints | $\frac{1}{3}n$ |
| Bounds | 0 |
| Linear equality constraints | 0 |
| Linear inequality constraints | 0 |
| Nonlinear equality constraints | $\frac{1}{3}n$ |
| Nonlinear inequality constraints | 0 |
| Nonzeros in $\nabla^2 f(x)$ | $n^2$ |
| Nonzeros in $c'(x)$ | $n$ |

### Performance

We provide results with the AMPL formulation on an SGI Onyx-2 Reality Monster. Results are summarized in Table 3.2. A quasi-uniform distribution of the point charges on a unit sphere was chosen as the standard starting guess for this problem.

The results in [10] show that most of the found solutions for $n_p \geq 110$ are not global (though SNOPT was able to find global minimizers for $n_p = 111, 115, 134, 138, 143, 149, 153$). The global solution for $n_p = 153$ is shown in Figure 3.1. We note that merit function evaluations are expensive, and that the Hessian is dense, which makes this problem computationally intensive and hard to solve for $n_p \geq 100$. LOQO was not able to find any solution for

Table 3.2: Performance of AMPL solvers

| Solver | $n_p = 50$ | $n_p = 75$ | $n_p = 100$ | $n_p = 150$ | $n_p = 200$ |
|---|---|---|---|---|---|
| DONLP2 | 14s | 88s | 497s | 1453s | 4911s |
| $f$ | 1055.182315 | 2454.369689 | 4448.350634 | 10236.43514 | 18438.92538 |
| $\|c(x)\|$ | 1.5423E-11 | 3.1587E-12 | 3.23075E-14 | 1.06827E-11 | 1.3968E-08 |
| iterations | 171 | 314 | 781 | 743 | 1141 |
| LANCELOT | 8s | 42s | 52s | 322s | 649s |
| $f$ | 1055.1823011 | 2454.369574 | 4448.350119 | 10236.26938 | 18438.99582 |
| $\|c(x)\|$ | 2.5892E-08 | 3.8406E-05 | 2.3241E-07 | 8.1215E-07 | 1.9401E-06 |
| iterations | 56 | 77 | 71 | 152 | 156 |
| MINOS | 20s | No | No | No | No |
| $f$ | 1055.1823147 | failure | failure | failure | failure |
| $\|c(x)\|$ | 1.2E-11 | | | | |
| iterations | 804 | | | | |
| SNOPT | 6s | 36s | 60s | 167s | 841s |
| $f$ | 1055.1823147 | 2454.369689 | 4448.350634 | 10236.25782 | 18439.32467 |
| $\|c(x)\|$ | 9.0E-12 | 2.6E-11 | 1.4E-11 | 6.7E-11 | 2.0E-11 |
| iterations | 357 | 748 | 722 | 817 | 1528 |
| LOQO | 125s | No | No | No | No |
| $f$ | 1056.604860 | failure | failure | failure | failure |
| dual $f$ | 1056.604851 | | | | |
| iterations | 335 | 10000 | 10000 | 10000 | 10000 |

$n_p \geq 75$, exceeding the iteration limit (stagnation or very slow progress toward a solution in all cases). MINOS could not solve the problem for $n_p \geq 75$ exiting with the message *Unbounded problem or bad initial guess*. DONLP2, LANCELOT and SNOPT, were able to find a local solution for all values of $n_p$ tried.

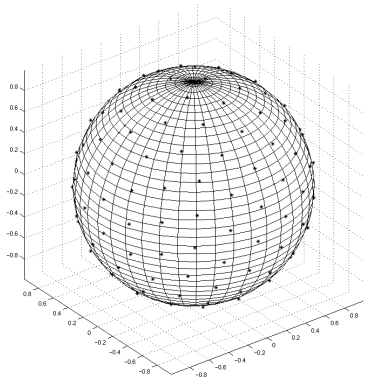

Figure 3.1: Optimal distribution of electrons on a conducting sphere, $n_p = 153$

## 4 Sawpath Tracking (Vanderbei [13])

Given a list of points $\{(x_i, y_i)\}_{i=0}^N$ describing the centerline of a wood piece, find the polynomial $p$ of degree at most $d$ that minimizes the difference between $\{y_i\}$ and $\{p(x_i)\}$ when $p$ satisfies the following constraints:

- the polynomial $p$ must go through the first point $(x_0, y_0)$ of the list;
- the initial slope of the polynomial $p$ must be $M$;
- the radius of curvature at every point must not exceed the radius $R$.

### Formulation

The merit function is

$$f(a) = \sum_{i=0}^{N} \left( \sum_{j=0}^{d} a_j x_i^j - y_i \right)^2$$

and the constraints are

$$\begin{aligned}
\sum_{j=0}^{d} a_j x_0^j &= y_0 \\
\sum_{j=1}^{d} j a_j x_0^{j-1} &= M \\
\left( R \sum_{j=2}^{d} j(j-1) a_j x_i^{j-2} \right)^2 &\le \left( 1 + \left( \sum_{j=1}^{d} j a_j x_i^{j-1} \right)^2 \right)^3, \qquad i = 0, 1, \ldots, N
\end{aligned}$$

We generalized this problem, as given in Vanderbei [13], from a polynomial of fourth degree to a polynomial of arbitrary degree $d$. In this formulation we followed [13] by modifying the curvature constraint to a constraint on the square of the radius.

Table 4.1: Sawpath tracking problem data

| | |
|---|---|
| Variables | $n = d + 1$ |
| Constraints | $N + 3$ |
| Bounds | 0 |
| Linear equality constraints | 2 |
| Linear inequality constraints | 0 |
| Nonlinear equality constraints | 0 |
| Nonlinear inequality constraints | $N + 1$ |
| Nonzeros in $\nabla^2 f(x)$ | $(d + 1)^2$ |
| Nonzeros in $c'(x)$ | $(N + 3)d + 1$ |

This problem has relatively few variables, but the presence of many nonlinear nonconvex inequality constraints makes it difficult to solve. If there are $d + 1$ distinct data points $x_i$, then $f$ is strictly convex and coercive. Thus, this problem has a unique solution.

### Performance

We provide results with the AMPL formulation on an SGI Onyx-2 Reality Monster. Results are summarized in Table 4.2 for the dataset from Vanderbei [13] with $N = 195$, $R = 2500$. Solutions for several values of $d$ are shown in Figure 4.1.

Table 4.2: Performance of AMPL solvers, $N = 195$

| Solver | $d = 2$ | $d = 3$ | $d = 4$ | $d = 5$ | $d = 6$ |
|---|---|---|---|---|---|
| DONLP2 | No | No | No | No | No |
| $f$ | 1152.737587 | 665.5803272 | 1091.265064 | 1023.788643 | 1194.409377 |
| $\|c(x)\|$ | 0.0E+00 | 1.81832E-06 | 2.77686E-15 | 2.13134E-09 | 0.0E+00 |
| iterations | 9 | 10 | 6 | 7 | 4 |
| LANCELOT | No | No | No | No | No |
| $f$ | failure | failure | failure | failure | failure |
| $\|c(x)\|$ | | | | | |
| iterations | 517 | 352 | 58 | 9 | 9 |
| MINOS | † | † | 1.4s | No | No |
| $f$ | 1152.706916 | 401.4899556 | 181.5729928 | ‡ | ‡ |
| $\|c(x)\|$ | 0.0E+00 | 1.6E-15 | 2.2E-16 | | |
| iterations | 6 | 18 | 87 | | |
| SNOPT | No | † | 11.3s | No | No |
| $f$ | failure | 401.4899555 | 181.5729928 | ‡ | ‡ |
| $\|c(x)\|$ | | 3.0E-14 | 2.2E-16 | | |
| iterations | 3717 | 1408 | 20512 | | |
| LOQO | † | † | 0.3s | 0.5s | 0.3s |
| $f$ | 1152.706890 | 401.4899495 | 181.5729922 | 151.2582871 | 64.72368379 |
| dual $f$ | 1152.706916 | 401.4899547 | 181.5729929 | 151.2582882 | 64.72369331 |
| iterations | 31 | 34 | 28 | 32 | 22 |

† Solution found in less than 0.1s ‡ Incorrect gradient or Jacobian

A major computational difficulty in this problem is the bad scaling when increasing $d$. The original data from Vanderbei [13] has data points $x_i$ ranging from 0 to 500, thus creating fairly bad scaling even for $d \geq 5$. LANCELOT iterates seemed to be diverging away from the solution even when the initial point was near the solution. MINOS and SNOPT gave warnings that the gradient of the objective and the Jacobian of the constraints were not correct and that the problem was not smooth (possible effects of the bad scaling). Yet, SNOPT and MINOS converged to a solution for $d = 2, 3, 4$, using gradients provided by AMPL. DONLP2 stopped prematurely with the message *relaxed KKT conditions satisfied*, or *unknown termination reason* for all $d$ tried. LOQO was able to find solutions for all $d$ tried ($d = 2, \ldots, 9$) in under 1 second. We also noticed that the problem becomes harder to solve as we increase the minimum radius of curvature $R$.

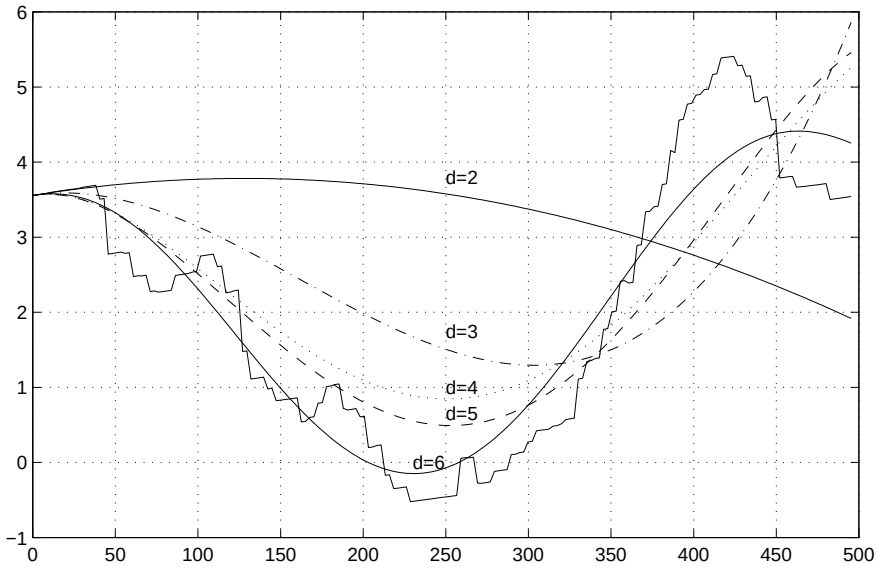


Figure 4.1: Solutions to the Sawpath Tracking problem for several $d$'s

## 5 Hanging Chain (H. Mittelmann, private communication)

Find the chain (of uniform density) of length $L$ suspended between two points with minimal potential energy.

### Implementation

This problem requires determining a function $x(t)$, the shape of the chain that minimizes the potential energy

$$\int_0^1 x\sqrt{1+x'^2}\,dt$$

subject to the constraint on the length of the chain,

$$\int_0^1 \sqrt{1+x'^2}\,dt = L,$$

and the end conditions $x(0) = a$ and $x(1) = b$. Discretization of this problem leads to an optimization problem with merit function

$$f(x) = h\sum_{i=1}^{n+1} \frac{x_i + x_{i-1}}{2}\sqrt{1+\left(\frac{x_i - x_{i-1}}{h}\right)^2}$$

and constraint

$$h\sum_{i=1}^{n+1} \sqrt{1+\left(\frac{x_i - x_{i-1}}{h}\right)^2} = L,$$

where $h = 1/(n+1)$, $x_0 = a$ and $x_{n+1} = b$.

Table 5.1: Hanging chain problem data

| | |
|---|---|
| Variables | $n$ |
| Constraints | 1 |
| Bounds | 0 |
| Linear equality constraints | 0 |
| Linear inequality constraints | 0 |
| Nonlinear equality constraints | 1 |
| Nonlinear inequality constraints | 0 |
| Nonzeros in $\nabla^2 f(x)$ | $3n-2$ |
| Nonzeros in $c'(x)$ | $n$ |

This problem has a nonconvex nonlinear merit function and one nonconvex nonlinear constraint. The solution to this problem seems to be unique.

### Performance

We provide results with the AMPL formulation on an SGI Onyx-2 Reality Monster. Results are summarized in Table 5.2 with $a = 1$, $b = 3$, $L = 4$. A piecewise linear chain of length $L$ was chosen as the standard starting guess. The solution for $n = 200$ is shown in Figure 5.1.

Table 5.2: Performance of AMPL solvers

| Solver | $n = 50$ | $n = 99$ | $n = 100$ | $n = 200$ | $n = 400$ |
|---|---|---|---|---|---|
| DONLP2 | 1s | 2s | 2s | 12s | 72s |
| $f$ | 5.068577962 | 5.068505564 | 5.068505062 | 5.068486411 | 5.068481694 |
| $\|c(x)\|$ | 2.84217E-14 | 4.54747E-13 | 1.13687E-13 | 0.0E+00 | 6.82121E-13 |
| iterations | 105 | 182 | 188 | 413 | 830 |
| LANCELOT | 8.9s | 33s | 46s | 190s | 1311s |
| $f$ | 5.068577968 | 5.068505567 | 5.068505065 | 5.068486411 | 5.068481697 |
| $\|c(x)\|$ | 1.5230E-08 | 4.9220E-08 | 5.3919E-08 | 2.9043E-07 | 3.6684E-07 |
| iterations | 1862 | 3472 | 4902 | 9054 | 25252 |
| MINOS | 0.5s | 2.3s | 2.4s | 11s | 55s |
| $f$ | 5.068577962 | 5.068505564 | 5.068505062 | 5.068486411 | 5.068481694 |
| $\|c(x)\|$ | 6.2E-10 | 9.4E-11 | 3.0E-11 | 1.6E-10 | 3.9E-10 |
| iterations | 293 | 557 | 572 | 1077 | 1948 |
| SNOPT | 0.9s | 10s | 77s | 150s | 2175s |
| $f$ | 5.068577962 | 5.068505564 | 5.068505063 | 5.068486413 | 5.068481697 |
| $\|c(x)\|$ | 6.3E-10 | 1.1E-09 | 1.1E-09 | 1.9E-09 | 4.2E-10 |
| iterations | 248 | 702 | 4205 | 1949 | 4667 |
| LOQO | No | No | No | No | No |
| $f$ | failure | failure | failure | failure | failure |
| $\|c(x)\|$ | | | | | |
| iterations | 553 | 920 | 1631 | 595 | 534 |

We noticed that SNOPT solved problems with $n$ odd much faster than problems with $n$ even. In general, DONLP2 and MINOS computed the solution much faster than SNOPT and LANCELOT in all cases. SNOPT was designed for the problems with few degrees of freedom in the constraints and in this problem the degrees of freedom grow linearly with the problem size $n$. Hence this behavior of SNOPT is expected. LOQO was unable to solve this problem even for $n \geq 50$. LOQO seems to converge to a solution and then suddenly diverges to a point far from the solution, declaring the problem infeasible.

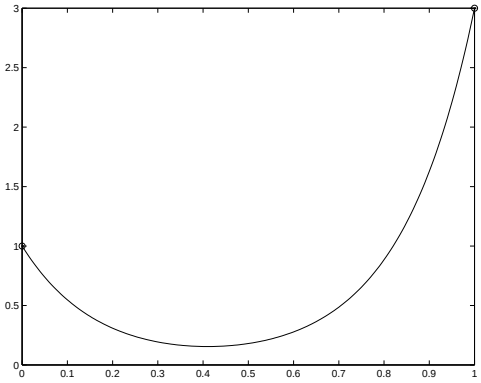


Figure 5.1: Hanging chain of length $L = 4$

## 6 Shape Optimization of a Cam (Anitescu and Serban [1])

Maximize the area of the valve opening for one rotation of a cam. The cam must be convex and the curvature of the cam must not exceed the curvature limit parameter $\alpha$. The radius of the cam must be between $R_{\min}$ and $R_{\max}$.

### Formulation

We assume that the shape of the cam is circular over an angle of $\frac{6}{5}\pi$ of its circumference, with radius $R_{\min}$. The design variables $r_i$, $i = 1, \ldots, n$ , represent the radius of the cam at equally spaced angles distributed over an angle of $\frac{2}{5}\pi$. $R_v$ is a design parameter related to the geometry of the valve.

Anitescu and Serban [1] show that the requirement that the cam be convex is equivalent to

$$-r_{i-1}r_i - r_i r_{i+1} + 2r_{i-1}r_{i+1}\cos(\Delta\theta) \leq 0, \quad i = 0, \ldots, n+1$$

where $r_{-1} = r_0 = R_{\min}$, $r_{n+1} = R_{\max}$, $r_{n+2} = r_n$ and $\Delta\theta = 2\pi/5(n+1)$. The curvature requirement is expressed by

$$\left(\frac{r_{i+1} - r_i}{\Delta\theta}\right)^2 \leq \alpha^2, \quad i = 0, \ldots, n$$

squaring the actual curvature constraints to make them smooth. The merit function is

$$f(r) = -\pi R_v^2 \sum_{i=1}^{n} r_i$$

and the constraints are

$$\begin{array}{rcll}
-R_{\min}^2 - R_{\min}r_1 + 2R_{\min}r_1\cos(\Delta\theta) & \leq & 0 & \\
-R_{\min}r_1 - r_1 r_2 + 2R_{\min}r_2\cos(\Delta\theta) & \leq & 0 & \\
-r_{i-1}r_i - r_i r_{i+1} + 2r_{i-1}r_{i+1}\cos(\Delta\theta) & \leq & 0 & i = 2, \ldots, n-1 \\
-r_{n-1}r_n - r_n R_{\max} + 2r_{n-1}R_{\max}\cos(\Delta\theta) & \leq & 0 & \\
-2R_{\max}r_n + 2r_n^2\cos(\Delta\theta) & \leq & 0 & \\
(r_1 - R_{\min})^2 - (\alpha\Delta\theta)^2 & \leq & 0 & \\
(r_{i+1} - r_i)^2 - (\alpha\Delta\theta)^2 & \leq & 0 & i = 1, \ldots, n-1 \\
(R_{\max} - r_n)^2 - (\alpha\Delta\theta)^2 & \leq & 0 & \\
R_{\min} \leq & r_i & \leq R_{\max} & i = 1, \ldots, n
\end{array}$$

Data for this problem appears in Table 6.1.

Since the optimal cam shape is symmetric, we only consider the half of the design angle. The problem was originally [1] formulated for the full angle of $4\pi/5$. This is a simple static model for the optimal shape design of a cam.

We used discretization with uniform angle partitions, which can be made more efficient with introducing angle partitions as variables as well. Introducing dynamic components into the model will complicate the problem and make it a lot harder to solve.

Table 6.1: Optimal design of a cam problem data

| | |
|---|---|
| Variables | $n$ |
| Constraints | $2n+3$ |
| Bounds | $n$ |
| Linear equality constraints | 0 |
| Linear inequality constraints | 1 |
| Nonlinear equality constraints | 0 |
| Nonlinear inequality constraints | $2n+2$ |
| Nonzeros in $\nabla^2 f(x)$ | 0 |
| Nonzeros in $c'(x)$ | $5n$ |

### Performance

We provide results with the AMPL formulation on an SGI Onyx-2 Reality Monster. Results are summarized in Table 6.2. Default values for the model constants were used: $R_{\min} = 1.0$, $R_{\max} = 2.0$, $R_v = 1.0$, $\alpha = 1.5$. We used a standard starting guess of $r_i = R_{\min}$, $i = 1, \ldots, n$, as suggested in [1]. Solutions for $n = 200$ with several values of $\alpha$ are shown in Figure 6.1.

Table 6.2: Performance of AMPL solvers

| Solver | $n = 10$ | $n = 50$ | $n = 100$ | $n = 200$ | $n = 400$ |
|---|---|---|---|---|---|
| DONLP2 | 2s | 14s | 438s | No † | No † |
| $f$ | -43.8599479 | -214.760855 | -428.4147433 | | |
| $\|c(x)\|$ | 1.04878E-08 | 2.26649E-07 | 3.70297E-07 | | |
| iterations | 12 | 156 | 576 | | |
| LANCELOT | 0.3s | 10s | ‡ | ‡ | ‡ |
| $f$ | -43.85989689 | -215.1835506 | -430.1620796 | -863.0490577 | -1810.253285 |
| $\|c(x)\|$ | 1.4863E-07 | 8.4264E-06 | 4.4204E-06 | 2.3595E-06 | 4.7278E-06 |
| iterations | 66 | 338 | 554 | 820 | 1121 |
| MINOS | 0.1s | No * | No * | No * | No * |
| $f$ | -43.85994780 | | | | |
| $\|c(x)\|$ | 4.4E-16 | 6.6E-01 | 1.1E-01 | 7.2E-02 | 7.9E-03 |
| iterations | 43 | 796 | 788 | 1583 | 1870 |
| SNOPT | 0.1s | 0.5s | 1.2s | 6.1s | No |
| $f$ | -43.85994884 | -214.758660 | -428.420412 | -855.700093 | -2436.977949 |
| $\|c(x)\|$ | 5.7E-08 | 1.3E-14 | 2.4E-05 | 5.3E-09 | 1.5E-02 |
| iterations | 43 | 728 | 1410 | 3735 | 12676 |
| LOQO | 0.1s | 0.8s | 4.8s | 20s | 84s |
| $f$ | -43.85994830 | -214.7608486 | -428.4147089 | -855.7000451 | -1710.275390 |
| dual $f$ | -43.85994844 | -214.7608470 | -428.4147221 | -855.7000511 | -1710.275397 |
| iterations | 40 | 168 | 386 | 534 | 704 |

† Problem is too large ‡ Step is too small * Infeasible problem

MINOS was not able to solve this problem for $n \geq 20$, exiting with the message *Infeasible problem (or bad starting guess)*. LANCELOT computed a shape very close to the optimal shape for $n \geq 100$, but stopped prematurely with the message *Step is too small*. Surprisingly, SNOPT did not solve the problem for $n = 400$ (stopped at an infeasible point with the exit condition *The current point cannot be improved*). SNOPT outperformed the other solvers

for smaller $n$. We note that the number of active constraints increased with $\alpha$ increasing up to a threshold of $\alpha_1 \approx 3.0$, after which increasing $\alpha$ did not change the optimal solution. The problem became harder to solve as we decreased $\alpha$ down to a threshold of $\alpha_0 \approx 1.25$, after which the problem was declared infeasible by all solvers.

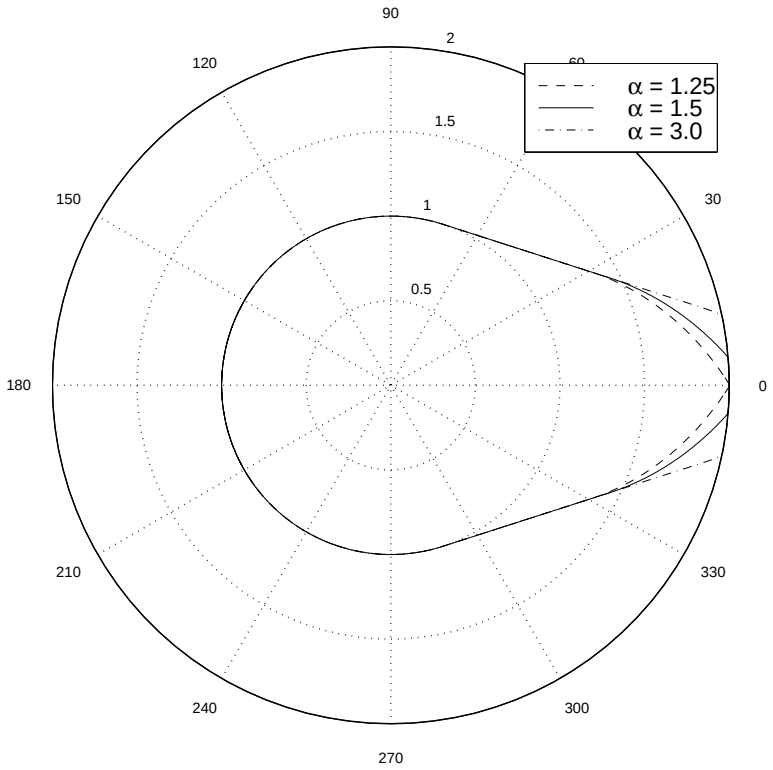


Figure 6.1: Cam shape with $n = 200$ and $\alpha = 1.25,\ 1.5,\ 3.0$.

## 7 Isometrization of $\alpha$-pinene (MINPACK-2 test problems [3])

Determine the reaction coefficients in the thermal isometrization of $\alpha$-pinene. The linear kinetic model proposed for this problem is

$$\begin{aligned} y_1' &= -(\theta_1+\theta_2)y_1 \\ y_2' &= \theta_1 y_1 \\ y_3' &= \theta_2 y_1 - (\theta_3+\theta_4)y_3 + \theta_5 y_5 \\ y_4' &= \theta_3 y_3 \\ y_5' &= \theta_4 y_3 - \theta_5 y_5 \end{aligned} \tag{7.1}$$

where $\theta_1, \ldots, \theta_5$ are the unknown coefficients. Initial conditions for (7.1) are known. Vectors of concentration measurements $z_j$ are given for $y$ at eight time points $\tau_1, \ldots, \tau_8$, where $y$ is the solution to (7.1). The $\alpha$-pinene problem is to minimize

$$\sum_{j=1}^{8} \|y(\tau_j;\theta) - z_j\|^2, \tag{7.2}$$

where $\theta$ is the vector with components $\theta_1, \ldots, \theta_5$ of unknown reaction coefficients. This formulation is based on the work of Box, Hunter, MacGregor, and Erjavac [5].

### Formulation

A $k$-stage collocation method approximates the solution of (7.1) by a vector-valued function $u : [0, t_f] \mapsto \mathbb{R}^5$, where each component of $u$ is a polynomial of order $k+1$ in each subinterval $[t_i, t_{i+1}]$ of a partition

$$0 = t_1 < t_2 < \cdots < t_{n_h} < t_{n_h+1} = t_f,$$

where $t_f \geq \tau_m$, and $\tau_m$ is the largest time measuremet. Thus $u$ is defined in terms of $5n_h(k+1)$ parameters. These parameters are determined by requiring that $u \in C[0, t_f]$ and that $u$ satisfy (7.1) at a set of $k$ collocation points in each interval $[t_i, t_{i+1}]$. We choose the collocation points $\rho_i$ as the roots of the $k$-th degree Legendre polynomial to guarantee superconvergence at the mesh points $t_i$.

Our formulation of the $\alpha$-pinene problem as an optimization problem follows [12, 3]. We use a uniform partitioning of the interval $[0, t_f]$ and the standard [2, pages 247–249] basis representation,

$$u_s(t) = v_{is} + \sum_{j=1}^{k} \frac{(t-t_i)^j}{j!\, h^{j-1}} w_{ijs}, \qquad t \in [t_i, t_{i+1}],$$

of the $s$-th component of the piecewise polynomial approximation $u$. The constraints in the optimization problem are the 5 initial conditions in (7.1), the continuity conditions, and the collocation equations. The continuity equations

$$u(t_{i+1}^-) = u(t_{i+1}^+), \qquad 1 \leq i < n_h,$$

are a set of $5(n_h - 1)$ linear equations. The collocation equations are a set of $5kn_k$ nonlinear equations obtained by requiring that $u$ satisfy (7.1) at the collocation points $\xi_{ij} = t_i + h\rho_j$ for $i = 1, \ldots, n_h$ and $j = 1, \ldots, k$.

Table 7.1: Isomerization of $\alpha$-pinene data

| | |
|---|---|
| Variables | $n = 25n_h + 5$ |
| Constraints | $25n_h$ |
| Bounds | 5 |
| Linear equality constraints | $5n_h$ |
| Linear inequality constraints | 0 |
| Nonlinear equality constraints | $20n_h$ |
| Nonlinear inequality constraints | 0 |
| Nonzeros in $\nabla^2 f(x)$ | $\leq 1600$ |
| Nonzeros in $c'(x)$ | $262n_h - 25$ |

This is a typical parameter estimation problem that arises in the modeling of physical phenomena with a parameter-dependent system of differential equations. We note that $n_h$ and $k$ can be specified, while other parameters are dependent on the problem. In our formulation we use $k = 4$. Arbitrarily large-dimensional test problems can be generated by selecting larger values of $n_h$. Note that this problem has only 5 degrees of freedom.

**Performance**

We provide results with the AMPL formulation on an SGI Onyx-2 Reality Monster. We used a starting point with zeros for the parameters and a piecewise constant approximation to (7.1) based on the linear interpolation of the measurement data onto the mesh points $t_i$. Results are summarized in Table 7.2. The solution for $n_h = 200$ is shown in Figure 7.1.

Table 7.2: Performance of AMPL solvers ($t_f = 37421$)

| Solver | $n_h = 20$ | $n_h = 50$ | $n_h = 100$ | $n_h = 150$ | $n_h = 200$ |
|---|---|---|---|---|---|
| LANCELOT | 182s † | 359s † | 1289s † | 2987s † | 6674s † |
| $\|c(x)\|$ | 1.7435E-06 | 3.7917E-06 | 7.9404E-06 | 3.3050E-06 | 1.8053E-06 |
| $f$ | 19.68852651 | 19.5489803 | 19.20467754 | 19.55026906 | 19.68888546 |
| iterations | 140 | 124 | 135 | 183 | 216 |
| MINOS | 13s | 10s | 52s | 134s | 230s |
| $f$ | 19.87208041 | 19.87216637 | 19.87216714 | 19.87216694 | 19.87216692 |
| $\|c(x)\|$ | 1.1E-12 | 5.0E-13 | 2.6E-10 | 1.9E-10 | 8.1E-11 |
| iterations | 531 | 780 | 1499 | 2129 | 2770 |
| SNOPT | 1.4s | 6.6s | 30s | 64s | 118s |
| $f$ | 19.87208041 | 19.87216637 | 19.87216700 | 19.87216697 | 19.87216696 |
| $\|c(x)\|$ | 9.7E-13 | 8.4E-13 | 6.9E-12 | 1.4E-11 | 1.4E-11 |
| iterations | 524 | 1301 | 2571 | 3897 | 5202 |
| LOQO | No | 116s | 2139s | 378s | 2054s |
| $f$ | failure | 19.87216631 | 19.87216641 | 19.87216637 | 19.87216692 |
| dual $f$ | | 19.87216636 | 19.87216695 | 19.87216649 | 19.87216686 |
| iterations | 10000 | 57 | 166 | 42 | 66 |

† Step is too small

DONLP2 stopped with message *relaxed KKT conditions satisfied: singular point* for smaller problems and was able to get a good fit to the data, but stopping short of the

optimal solution. Since the problems were too large for DONLP2 when $n_h > 28$, we did not include DONLP2 in Table 7.2.

LANCELOT stopped with message *step is too small*, very near the solution for all $n_h$ we tried (projected gradient norm was on the order of $10^{-4}$ for $n_h = 100, 150, 200$ with default optimality tolerance of $10^{-5}$). Parameters estimated by LANCELOT were fairly accurate compared with the parameters obtained with SNOPT. MINOS and SNOPT were able to solve the problem for all $n_h$ tried, but SNOPT was more efficient by about a factor of 2. LOQO was not able to solve problems with small $n_h$, but the performance improved for larger $n_h$. LOQO was slower than MINOS and SNOPT. In the iteration log of LOQO the message *dependent rows* appeared often near the solution, which might explain the degraded performance. All solvers were able to estimate reaction parameters with enough accuracy for practical purposes.

The choice of the final time $t_f$ had a significant effect on the performance of the solvers. As $t_f$ increased, the problem became harder to solve and performance of all solvers degraded. In some cases, LOQO and MINOS were not able to solve the problem at all.

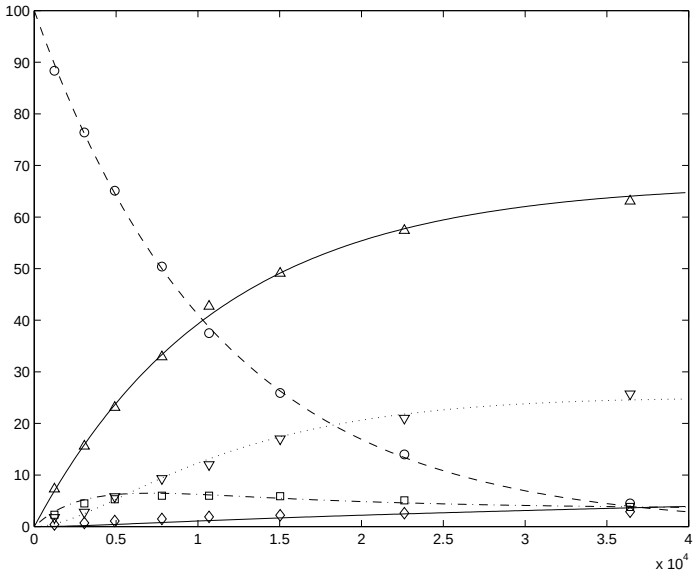


Figure 7.1: The five components of $u(t, \theta)$ for the $\alpha$-pinene problem with the optimal $\theta$

## 8 Marine Population Dynamics (Rothschild *et al* [11])

Given estimates of the abundance of the population of a marine species at each stage (for example, nauplii, juvenile, adult) as a function of time, determine stage specific growth and mortality rates. The model for the population dynamics of the $n_s$-stage population (for the short time periods) used in [11] is

$$y_j' = g_{j-1}y_{j-1} - (m_j + g_j)y_j, \qquad 1 \le j \le n_s, \tag{8.1}$$

where $m_i$ and $g_i$ are the unknown mortality and growth rates at stage $i$ with $g_0 = g_{n_s} = 0$. This model assumes that the species eventually dies or grows into the next stage, with the implicit assumption that the species cannot skip a stage. Initial conditions for the differential equations are unknown, since the stage abundance measurements at the initial time might also be contaminated with experimental error. We minimize the error between computed and observed data,

$$\sum_{j=1}^{n_m} \|y(\tau_j; m, g) - u_j\|^2,$$

where $m$ and $g$ are, respectively, vectors of mortality and growth rates with components $m_1, \ldots, m_{n_s}$ and $g_1, \ldots, g_{n_s-1}$, and $n_m$ is the number of the stage abundance measurements.

### Formulation

We use a $k$-stage collocation method to formulate this problem as an optimization problem. In this approach the solution to (8.1) is represented by a vector-valued function $u : [0, t_f] \mapsto \mathbb{R}^{n_s}$, where each component of $u$ is a polynomial of order $k+1$ in each subinterval $[t_i, t_{i+1}]$ of a partition of $[0, t_f]$, where $t_f \ge \tau_{n_m}$ and $\tau_{n_m}$ is the largest time measurement. We use a uniform partitioning of $[0, t_f]$, and the standard [2, pages 247–249] basis representation,

$$u_s(t) = v_{is} + \sum_{j=1}^{k} \frac{(t - t_i)^j}{j!\, h^{j-1}} w_{ijs}, \qquad t \in [t_i, t_{i+1}],$$

of the $s$-th component of $u$. The constraints in the optimization problem are the continuity conditions and the collocation equations. The continuity equations are a set of $n_s(n_h - 1)$ linear equations. The collocation equations are a set of $k\, n_s\, n_k$ nonlinear equations obtained by requiring that $u$ satisfy (8.1) at the collocation points $\xi_{ij} = t_i + h\rho_j$ for $i = 1, \ldots, n_h$ and $j = 1, \ldots, k$.

The parameters in the optimization problem are the $n_s n_h$ initial conditions, the $n_s$ mortality rates, the $n_s - 1$ growth rates, and the $k n_s n_h$ parameters $w_{ijk}$ in the representation of $u$. Data for this problem, with $k = 4$, appears in Table 8.1.

We do not impose any initial conditions on the differential equations since initial measurements are usually contaminated with experimental error. Introducing these extra degrees of freedom into the problem formulation should allow solvers to find a better fit to the data. A significant difference between this problem and the $\alpha$-pinene is that the population dynamics data usually contains large observation errors.

Table 8.1: Marine population dynamics problem data

| | |
|---|---|
| Variables | $n = 5n_s n_h + 2n_s - 1$ |
| Constraints | $5n_s n_h - n_s$ |
| Bounds | $2n_s - 1$ |
| Linear equality constraints | $n_s(n_h - 1)$ |
| Linear inequality constraints | 0 |
| Nonlinear equality constraints | $4n_s n_h$ |
| Nonlinear inequality constraints | 0 |
| Nonzeros in $\nabla^2 f(x)$ | $\leq (n_s n_m)^2$ |
| Nonzeros in $c'(x)$ | $58n_s n_h - 28n_h - 6n_s$ |

**Performance**

We provide results with the AMPL formulation on an SGI Onyx-2 Reality Monster. We used a simulated dataset with $n_s = 8$ stages. We used a standard initial starting point with zeros for the parameters and a piecewise constant approximation to the solution of (8.1) based on the linear interpolation of the measurement data onto the mesh points $t_i$. Results are summarized in Table 8.2. We used the default options for the solvers, except for setting iteration and variable limits high enough for the problem size. The solution for $n_s = 8$ is shown in Figure 8.1.

Table 8.2: Performance of AMPL solvers

| Solver | $n_h = 25$ | $n_h = 50$ | $n_h = 100$ | $n_h = 150$ | $n_h = 200$ |
|---|---|---|---|---|---|
| LANCELOT | † | † | † | † | † |
| $f$ | 19746526.87 | 19746529.70 | 19746529.24 | 19746528.57 | 19746529.52 |
| $\|c(x)\|$ | 2.8021E-06 | 1.8881E-06 | 5.4092E-06 | 8.6955E-06 | 1.1871E-06 |
| iterations | 556 | 283 | 276 | 289 | 332 |
| MINOS | 20s | No ‡ | No ‡ | No | No |
| $f$ | 19746526.83 | 90765626.58 | 38788064.26 | 298683521.2 | 520276498.4 |
| $\|c(x)\|$ | 8.0E-12 | 3.7E-11 | 6.8E-11 | 1.0E-05 | 1.0E-01 |
| iterations | 1058 | 1780 | 2031 | | |
| SNOPT | 14s | 28s | 79s | 209s | 479s |
| $f$ | 19746526.83 | 19746529.71 | 19746529.72 | 19746529.72 | 19746529.72 |
| $\|c(x)\|$ | 5.2E-12 | 4.5E-12 | 3.4E-12 | 1.1E-11 | 1.1E-11 |
| iterations | 1652 | 2672 | 4795 | 6915 | 9507 |
| LOQO | No | No | No | No | No |
| $f$ | failure | failure | failure | failure | failure |
| dual $f$ | | | | | |
| iterations | 10000 | 968 | 10000 | 719 | 758 |

† Step is too small ‡ Possibly a local minimizer

This problem was too large for DONLP2 even with $n_h = 20$, so results are not included for DONLP2. LANCELOT found solutions for all $n_h$. It is interesting to note that LANCELOT used about 10 times more memory to solve this problem than other solvers. MINOS solved the problem for $n_h = 25$, but stopped at a suboptimal point for other $n_h$ tried. For $n_h = 50, 100$ MINOS claimed to stop at an optimal point. For $n_h = 150, 200$,

MINOS stopped with the message *the current point cannot be improved* at a suboptimal point. SNOPT successfully found solutions for all $n_h$. LOQO did not solve the problem for any $n_h$, either running over the iterations limit with no significant progress towards a solution or stopping with the message *primal or dual infeasible*. In the iteration log of LOQO the message *dependent rows* appeared often near the solution, which might explain the degraded performance of LOQO.

As in the $\alpha$-pinene problem, we noticed that performance is slightly sensitive to the choice of the final time $t_f$. Choosing $t_f$ very close to the last measurement time $\tau_{n_m}$ made the problem easiest to solve, but LOQO or MINOS still could not solve the problem.

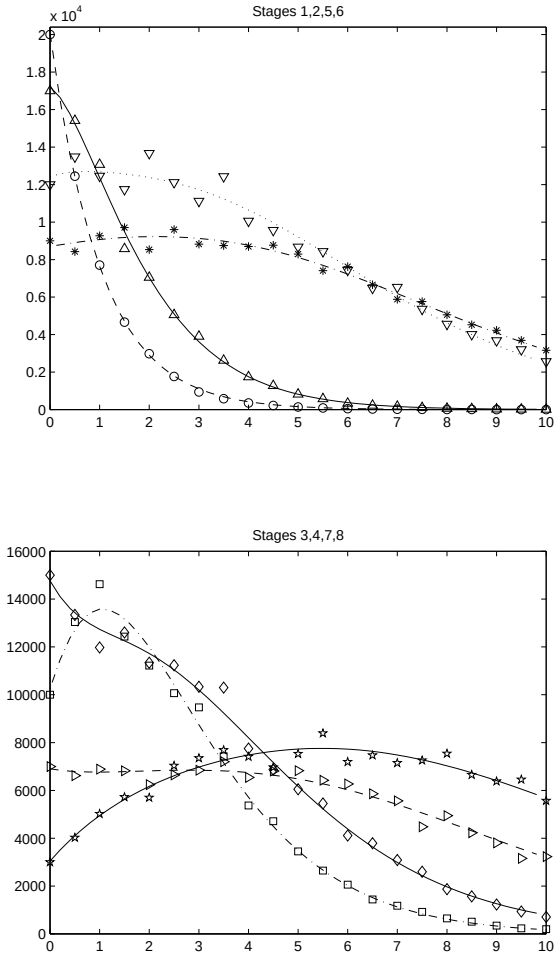


Figure 8.1: $u$ with optimal mortality and growth parameters for $n_s = 8$.

## 9 Flow in a Channel (MINPACK-2 test problems [3])

Analyze the flow of a fluid during injection into a long vertical channel, assuming that the flow is modeled by the boundary value problem,

$$\begin{aligned} &u'''' = R\,(u'u'' - uu'''), \qquad 0 \le t \le 1, \\ &u(0) = 0, \quad u(1) = 1, \quad u'(0) = u'(1) = 0. \end{aligned} \tag{9.1}$$

where $u$ is the potential function, $u'$ is the tangential velocity of the fluid, and $R$ is the Reynolds number.

### Formulation

We use a $k$-stage collocation method to formulate this problem as an optimization problem with a constant merit function and equality constraints representing the solution of (9.1).

We approximate the solution of (9.1) by a piecewise polynomial $u$. We use a uniform partitioning $\{t_i\}$ of $[0,1]$, and the standard [2, pages 247–249] basis representation,

$$u(t) = \sum_{j=1}^{m} \frac{(t-t_i)^{j-1}}{(j-1)!} v_{ij} + \sum_{j=1}^{k} \frac{(t-t_i)^{j+m-1}}{(j+m-1)!\,h^{j-1}} w_{ij}, \qquad t \in [t_i, t_{i+1}]$$

for $u$. Note that $u \in C^{m-1}[0,1]$, where $m = 4$ is the order of the differential equation.

The constraints in the optimization problem are the initial conditions in (9.1), the continuity conditions and the collocation equations. There are $m = 4$ initial conditions. The continuity equations are a set of $m(n_h - 1)$ linear equations. The collocation equations are a set of $k\,n_h$ nonlinear equations obtained by requiring that $u$ satisfy (8.1) at the collocation points $\xi_{ij} = t_i + h\rho_j$ for $i = 1, \ldots, n_h$ and $j = 1, \ldots, k$. The collocation points $\rho_j$ are the roots of the $k$-th degree Legendre polynomial.

Table 9.1: Flow in a channel problem data

| | |
|---|---|
| Variables | $n = 8n_h$ |
| Constraints | $8n_h$ |
| Bounds | 0 |
| Linear equality constraints | $4n_h$ |
| Linear inequality constraints | 0 |
| Nonlinear equality constraints | $4n_h$ |
| Nonlinear inequality constraints | 0 |
| Nonzeros in $\nabla^2 f(x)$ | 0 |
| Nonzeros in $c'(x)$ | $62n_h - 13$ |

The parameters in the optimization problem are the $(m+k)n_h$ parameters $v_{ij}$ and $w_{ij}$ in the representation of $u$. Data for this problem, with $k = 4$, appears in Table 9.1. This problem is easy to solve for small Reynolds numbers but becomes increasingly difficult to solve as $R$ increases.

**Performance**

We provide results with the AMPL formulation on a Sun UltraSPARC2. For $R = 10$, we used the solution of the boundary value problem (9.1) for $R = 0$, as the starting point for all solvers. For larger values of the Reynolds number we used continuation. Results are summarized in Tables 9.2 and 9.3. We used the default options for the solvers, except for setting iteration and variable limits high enough for the problem size. Solutions for several $R$ with $n_h = 200$ are shown in Figure 9.1:

Table 9.2: Performance of AMPL solvers

| | $n_h = 40$ | | | | $n_h = 100$ | | | |
|---|---|---|---|---|---|---|---|---|
| Solver | $R = 10$ | $R = 10^2$ | $R = 10^3$ | $R = 10^4$ | $R = 10$ | $R = 10^2$ | $R = 10^3$ | $R = 10^4$ |
| LANCELOT | No | No | No | No | No | No | No | No |
| iterations | 10000 | 10000 | 10000 | 10000 | 10000 | 10000 | 10000 | 10000 |
| $\|c(x)\|$ | | | | | | | | |
| MINOS | 0.8s | 0.2s | 0.2s | 0.3s | 3.7s | 0.6s | 0.6s | 0.6s |
| iterations | 178 | 5 | 5 | 6 | 442 | 5 | 5 | 6 |
| $\|c(x)\|$ | 5.4E-13 | 4.5E-13 | 2.9E-11 | 1.2E-08 | 2.4E-13 | 9.1E-13 | 2.9E-11 | 1.2E-09 |
| SNOPT | 1.7s | 58s | 2.7s | No | 8.4s | 42s | No | No |
| iterations | 358 | 1582 | 457 | failure | 846 | 1418 | failure | failure |
| $\|c(x)\|$ | 2.7E-09 | 9.1E-13 | 2.2E-11 | 5.3E+05 | 2.7E-09 | 1.1E-12 | 1.9E+04 | 2.0E+05 |
| LOQO | 1.0s | No | No | No | 5.4s | No | No | No |
| iterations | 28 | 10000 | 10000 | 10000 | 32 | 10000 | 10000 | 10000 |
| duality gap | 2.0E-08 | | | | 1.66E-08 | | | |

Table 9.3: Performance of AMPL solvers

| | $n_h = 200$ | | | | $n_h = 400$ | | | |
|---|---|---|---|---|---|---|---|---|
| Solver | $R = 10$ | $R = 10^2$ | $R = 10^3$ | $R = 10^4$ | $R = 10$ | $R = 10^2$ | $R = 10^3$ | $R = 10^4$ |
| LANCELOT | No | No | No | No | No | No | No | No |
| iterations | 10000 | 10000 | 10000 | 10000 | 10000 | 10000 | 10000 | 10000 |
| $\|c(x)\|$ | | | | | | | | |
| MINOS | 14s | 1.2s | 1.2s | 1.4s | 46s | 2.6s | 3.0s | 7.0s |
| iterations | 884 | 5 | 5 | 6 | 1560 | 5 | 5 | 109 |
| $\|c(x)\|$ | 1.8E-13 | 9.1E-13 | 2.2E-11 | 4.7E-10 | 3.8E-07 | 2.5E-08 | 2.5E-09 | 1.1E-08 |
| SNOPT | 31s | 67s | No | No | 115s | No | No | No |
| iterations | 1675 | 2226 | failure | failure | 3112 | failure | failure | failure |
| $\|c(x)\|$ | 2.7E-09 | 5.0E-07 | 1.8E+04 | 2.0E+05 | 3.8E-07 | 9.2E+02 | 9.9E+03 | 1.0E+05 |
| LOQO | 25s | No | No | No | 50s | No | No | No |
| iterations | 44 | 10000 | 10000 | 10000 | 34 | 10000 | 10000 | 10000 |
| duality gap | 7.0E-09 | | | | 2.5E-08 | | | |

LANCELOT was not able to solve even simple version of the problem, advancing very slowly towards the solution (as judged from the value of the merit function), and running over the iteration limit. MINOS was very successful on this problem, obtaining solutions for all values of $R$ and $n_h$ tried, and outperforming SNOPT by at least a factor of 2 in all cases. SNOPT solved the problem for $R = 10, 10^2, 10^3$ when $n_h = 40$, but performance

degraded with increasing $n_h$, and for $n_h = 400$, SNOPT could not find a solution even for $R = 10^2$. LOQO was able to solve the problem for $R = 10$ for all values of $n_h$ tried, but failed to converge for larger values of $R$ in all cases, with dual objective slowly increasing to a large positive number. We also note that MINOS was able to find a solution from the standard initial point for all values of $R$ in the range from 0 to $10^5$.

We also used SNOPT with an F77 implementation of this problem. In this set of experiments we used the solution of (9.1) for $R = 0$ as the starting point for $R = 10, 10^2$, and used the solution of the problem for $R = 10^2$ as starting point for higher Reynolds numbers. The results are summarized in Table (9.4).

Table 9.4: Performance of SNOPT with F77 code

| | $n_h = 50$ | $n_h = 100$ | $n_h = 200$ | $n_h = 400$ |
|---|---|---|---|---|
| $R = 10$ | 2.6s | 9.6s | 44s | 196s |
| $R = 10^2$ | 21s | 35s | 237s | 864s |
| $R = 10^3$ | 1311s | Not solved | 2077s | 4304s |
| $R = 10^4$ | Not solved | 3863s | 16907s | 35927s |

The results in Table 9.4 are not comparable with those in Tables 9.2-9.3 because we used different starting points, but we noted improved global convergence. The difference in behavior may be partially explained by the fact that we did not separate linear and nonlinear constraints in the F77 implementation.

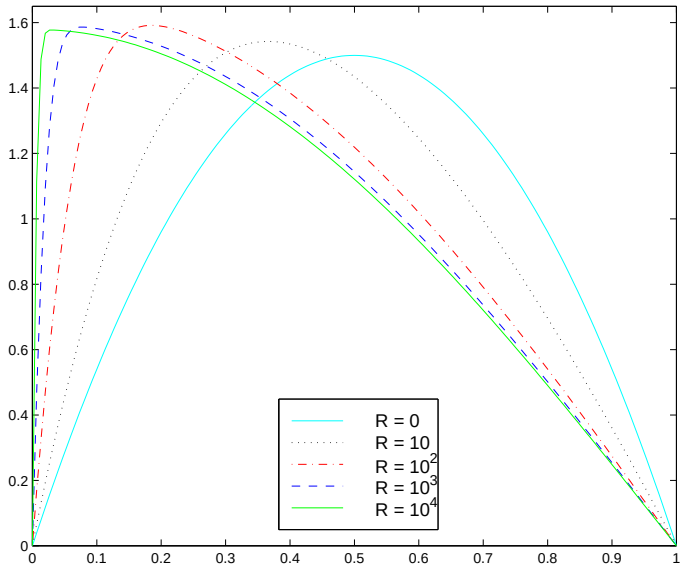


Figure 9.1: Solutions for $R = 0, 10, 10^2, 10^3, 10^4$ with $n_h = 200$

## 10 Non-inertial Robot Arm (Vanderbei [13])

Minimize the time taken for a robot arm to move from one point to another while satisfying boundary conditions, path constraints, and physical laws.

### Formulation

The arm is a rigid bar of length $L$ that protrudes a distance $\rho$ from the origin to the gripping end and sticks out a distance $L - \rho$ in the opposite direction. If the pivot point of the arm is the origin of a spherical coordinate system, then the problem can be phrased in terms of

$$\begin{array}{rcl} \rho(t) & \equiv & \text{length of arm from pivot} \\ \theta(t) & \equiv & \text{angle in horizontal plane} \\ \phi(t) & \equiv & \text{angle in vertical plane} \\ u_\rho(t), u_\theta(t), u_\phi(t) & \equiv & \text{controls in basis directions} \\ t_f & \equiv & \text{final time} \end{array}$$

Bounds on the variables are

$$\begin{array}{rcccl} & 0 & \leq & t_f & \\ 0 & \leq & \rho(t) & \leq & L \\ -\pi & \leq & \theta(t) & \leq & \pi \\ 0 & \leq & \phi(t) & \leq & \pi \\ -\bar{u}_\rho & \leq & u_\rho(t) & \leq & \bar{u}_\rho \\ -\bar{u}_\theta & \leq & u_\theta(t) & \leq & \bar{u}_\theta \\ -\bar{u}_\phi & \leq & u_\phi(t) & \leq & \bar{u}_\phi \end{array} \tag{10.1}$$

where $\bar{u}_\rho$, $\bar{u}_\theta$, and $\bar{u}_\phi$ are the most extreme controls allowed. The controls $u$ are applied in the coordinate directions and therefore they enter the system as the constraints

$$L\ddot{\rho} = u_\rho, \qquad I_\theta \ddot{\theta} = u_\theta, \qquad I_\phi \ddot{\phi} = u_\phi \tag{10.2}$$

where $I$ is the moment of inertia, defined by

$$I_\theta = \frac{((L-\rho)^3 + \rho^3)}{3} \sin(\phi)^2, \qquad I_\phi = \frac{((L-\rho)^3 + \rho^3)}{3}.$$

The boundary conditions are

$$\rho(0) = 4.5,\ \rho(t_f) = 4.5, \qquad \theta(0) = 0,\ \theta(t_f) = \frac{2\pi}{3}, \qquad \phi(0) = \frac{\pi}{4},\ \phi(t_f) = \frac{\pi}{4}$$

$$\dot{\rho}(0) = \dot{\theta}(0) = \dot{\phi}(0) = \dot{\rho}(t_f) = \dot{\theta}(t_f) = \dot{\phi}(t_f) = 0$$

This model ignores the fact that the spherical coordinate reference frame is a non-inertial frame and should have terms for coriolis and centrifugal forces.

### Implementation I

In this implementation, the controls $u$ are eliminated by substitution. Therefore, the equality constraints in (10.2) become the inequalities

$$\begin{array}{rcccl} -\bar{u}_\rho & \leq & L\ddot{\rho} & \leq & \bar{u}_\rho \\ -\bar{u}_\theta & \leq & I_\theta\ddot{\theta} & \leq & \bar{u}_\theta \\ -\bar{u}_\phi & \leq & I_\phi\ddot{\phi} & \leq & \bar{u}_\phi \end{array}$$

Discretization of the problem involved using a uniform time step and introducing new variables representing the first and second derivatives of the state variables. New constraints were introduced requiring the new variables satisfy first order difference approximations to the derivatives. The number of grid points at which the state variables are evaluated is $N$. The velocities, accelerations, and moments are evaluated at slightly fewer grid points. The variable in the optimization problem are

$$\rho(1:N),\ \dot{\rho}(1:N-1),\ \ddot{\rho}(1:N-2), \qquad \theta(1:N),\ \dot{\theta}(1:N-1),\ \ddot{\theta}(1:N-2),$$

$$\phi(1:N),\ \dot{\phi}(1:N-1),\ \ddot{\phi}(1:N-2), \qquad I_\theta(1:N-2),\ I_\phi(1:N-2), \qquad t_f$$

In this problem, $\bar{u}_\rho = \bar{u}_\theta = \bar{u}_\phi = 1$, and $L = 5$. Problem data appears in Table 10.1.

Table 10.1: Non-inertial robot arm problem data

| | |
|---|---|
| Variables | $11N - 12$ |
| Constraints | $10N - 5$ |
| Bounds | $4N - 2$ |
| Linear equality constraints | 12 |
| Linear inequality constraints | 0 |
| Nonlinear equality constraints | $8N - 13$ |
| Nonlinear inequality constraints | $2N - 4$ |
| Nonzeros in $\nabla^2 f(x)$ | 0 |
| Nonzeros in $c'(x)$ | $29N - 36$ |

### Performance

We provide results with the AMPL formulation on a Sun UltraSPARC2. All of the solvers were given the same initial values as suggested by Vanderbei [13]. The initial values for the state variables are straight lines for the first half of the interval and parabolas for the second half. Difference approximations were given as guesses for the derivative variables. The initial values for the moments of inertia were based upon difference approximations to the second derivatives, while the initial value for the final time was $t_f = 1000/N$.

Table 10.2 shows the computational results for various values of $N$. MINOS is unable to solve this problem for $N = 50, 100$. However, aside from these two instances, the rest of the solvers seem to converge to the correct solution for all $N$.

Figures 10.1 and 10.2 show the optimal path of the robot arm for $N = 100$ as calculated by MINOS (using the implementation II), while Figure 10.3 shows each of the variables

Table 10.2: Performance of AMPL solvers (Implementation I)

| Solver | $N = 10$ | $N = 50$ | $N = 100$ | $N = 500$ |
|---|---|---|---|---|
| LANCELOT | Yes | Yes | Yes | Yes |
| $t_f$ | 10.31945331 | 9.331494417 | 9.234452773 | 9.159271883 |
| iters \| sec | 104 \| 3.35 | 138 \| 70.01 | 173 \| 250.62 | 343 \| 276.39 |
| MINOS | Yes | infeasible | Yes | Yes |
| $t_f$ | 10.3194546 | - | 9.234453135 | 9.159271891 |
| iters \| sec | 450 \| 0.98 | 4848 \| 23.37 | 851 \| 12.98 | 3101 \| 313.17 |
| SNOPT | Yes | Yes | Yes | Yes |
| $t_f$ | 10.3194546 | 9.331495269 | 9.234453135 | 9.159271887 |
| iters \| sec | 967 \| 3.20 | 4006 \| 31.87 | 7313 \| 128.28 | 46943 \| 112.36 |
| LOQO | Yes | Yes | Yes | Yes |
| $t_f$ | 10.31945462 | 9.331495269 | 9.234453135 | 9.159271891 |
| iters \| sec | 24 \| 0.23 | 35 \| 4.33 | 58 \| 32.55 | 359 \| 775.743 |

individually. Note that the controls are calculated from the other known variables. The paths reported by the solvers are all identical (assuming they reported finding an optimal point), thus only one graph is shown.

### Implementation II

In this implementation the moments $(I_\theta, I_\phi)$ were eliminated by substitution. Discretization of the problem involved using a uniform time step for the integration of (10.2) over $N$ grid points. The variables in the optimization problem are

$$\rho(1:N),\ \dot{\rho}(1:N), \qquad \theta(1:N),\ \dot{\theta}(1:N), \qquad \phi(1:N),\ \dot{\phi}(1:N),$$

$$u_\rho(1:N),\ u_\theta(1:N),\ u_\phi(1:N), \qquad t_f$$

In this problem $\bar{u}_\rho = \bar{u}_\theta = \bar{u}_\phi = 1$ and $L = 5$. Data for this problem is shown in Table 10.3.

Table 10.3: Non-inertial robot arm problem data (Implementation II)

| | |
|---|---|
| Variables | $9N + 1$ |
| Constraints | $6(N-1) + 12$ |
| Bounds | $7N + 1$ |
| Linear equality constraints | 12 |
| Linear inequality constraints | 0 |
| Nonlinear equality constraints | $6(N-1)$ |
| Nonlinear inequality constraints | 0 |
| Nonzeros in $\nabla^2 f(x)$ | 0 |
| Nonzeros in $c'(x)$ | $36(N-1) + 12$ |

### Performance

We provide results with the AMPL formulation on a Sun UltraSPARC2. In addition, a C version was also implemented for SNOPT, with the derivatives generated by ADIC, thus

allowing a comparison between the AMPL version and the ADIC augmented C version.

Table 10.4: Performance of AMPL solvers (Implementation II)

| Solver | $N = 10$ | $N = 50$ | $N = 100$ | $N = 500$ |
|---|---|---|---|---|
| LANCELOT | no feasible solution | no feasible solution | iteration limit | no feasible solution |
| $t_f$ | 0 | 0 | 0 | 0 |
| iters \| sec | 3 \| 0.08 | 44 \| 15.71 | 1000 \| 139.56 | - |
| MINOS | Yes | Yes | Yes | Yes |
| $t_f$ | 9.27862977 | 9.145749287 | 9.141994656 | 9.141334372 |
| iters \| sec | 87 \| 0.21 | 390 \| 3.48 | 473 \| 11.24 | 1634 \| 305.20 |
| SNOPT | Yes | Yes | infeasible | infeasible |
| $t_f$ | 9.27862977 | 9.145749287 | - | - |
| iters \| sec | 875 \| 2.30 | 11500 \| 64.13 | 2177 \| 10.64 | 14081 \| 315.56 |
| LOQO | infeasible | iteration limit | iteration limit | iteration limit |
| $t_f$ | - | - | - | - |
| iters \| sec | 463 \| 13.72 | 1000 \| 154.15 | 1000 \| 193.72 | - |

All solvers were given the same initial values. Where possible, straight lines between the boundary conditions or (in the absence of boundary conditions) zeros were given as initial values. The exceptions are for $t_f$ which was set to 1000, and for $\theta$ which was initialized to a parabola passing through $(0,0)$, $(0.5,1)$, $(1,0)$. If $\theta$ is not initialized in this manner, SNOPT considers the problem infeasible.

Table 10.4 shows the computational results for various values of $N$. Note that while the alternative implementation is faster, fewer of the solvers converge to the correct solution. However, it is interesting to note that for this implementation, the solvers that did find the correct solution did so in considerably less time than it took using the first implementation.

Table 10.5: Performance of SNOPT with C implementation

| Solver | $N = 10$ | $N = 50$ | $N = 100$ | $N = 500$ |
|---|---|---|---|---|
| SNOPT | infeasible problem | Yes | Yes | Yes |
| $t_f$ | - | 9.1457563370858 | 9.1420016949989 | 9.1409521227122 |
| iters | 120 | 1937 | 4981 | 15713 |
| constraint (sec) | 0.03 | 0.98 | 1.99 | 8.33 |
| objective (sec) | 0.00 | 0.07 | 0.13 | 0.07 |
| solve (sec) | 0.05 | 6.72 | 21.44 | 1835.08 |

Again, Figures 10.1, 10.2, and 10.3 show the optimal path of the robot arm for $N = 100$, calculated using MINOS. Figures generated from the output from the C version are not shown, since they are also identical to the alternative AMPL/MINOS version.

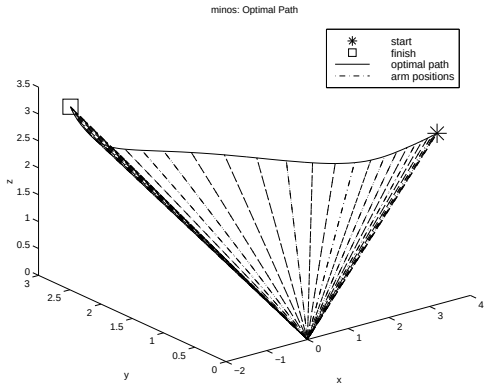


Figure 10.1: Non-inertial Robot Arm Optimal Path (side view)

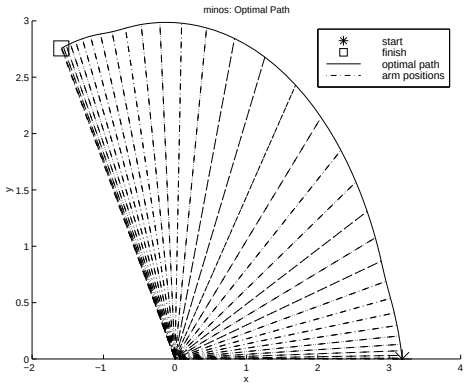


Figure 10.2: Non-inertial Robot Arm Optimal Path (top view)

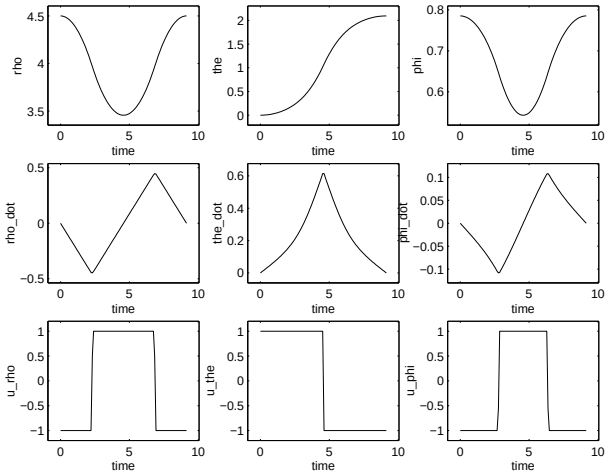


Figure 10.3: Non-intertial Robot Arm Optimal Path (individual variables)

## 11 Linear Tangent Steering (Betts, Eldersveld, Huffman [4])

Minimize the time taken for a point mass, acted upon by a thrust of constant magnitude, to satisfy boundary conditions, path constraints, and the differential equations governing motion to pass from one point to another.

### Formulation

The behavior of a point mass acted upon by a force of magnitude $a$ can be modeled using the system of second order differential equations,

$$\begin{aligned} \ddot{y}_1 &= a\cos(u) \\ \ddot{y}_2 &= a\sin(u) \end{aligned} \tag{11.1}$$

where

$$\begin{aligned} y_1(t) &\equiv \text{first position coordinate} \\ y_2(t) &\equiv \text{second position coordinate} \\ u(t) &\equiv \text{control angle} \\ t_f &\equiv \text{final time} \end{aligned}$$

and $a$ is the constant magnitude of thrust. In this case, $a = 100$. Bounds on the variables are

$$t_f \geq 0, \qquad -\frac{\pi}{2} \leq u(t) \leq \frac{\pi}{2}.$$

The constraints are (11.1). The boundary conditions, as given in [4], are

$$y_1(0) = y_2(0) = \dot{y}_1(0) = \dot{y}_2(0) = 0, \qquad y_2(t_f) = 5, \quad \dot{y}_1(t_f) = 45, \quad \dot{y}_2(t_f) = 0.$$

System (11.1) can be expressed as the system of four first order differential equations

$$\begin{aligned} \dot{y}_1 &= y_3 \\ \dot{y}_2 &= y_4 \\ \dot{y}_3 &= a\cos(u) \\ \dot{y}_4 &= a\sin(u) \end{aligned} \tag{11.2}$$

where $y_3$ and $y_4$ are the velocity coordinates of the point mass.

Discretization was done using a uniform time step and the trapezoidal rule for the integration of the system over $N$ grid points. By treating the final time $t_f$ as the objective function to be minimized, and the trapezoidal discretization and bounds on $u$ as constraints, we can formulate the problem as an optimization problems with variables

$$y_1(1:N), \quad y_2(1:N), \quad y_3(1:N), \quad y_4(1:N), \quad u(1:N), \quad t_f.$$

Data for this problem is shown in Table 11.1.

Table 11.1: Linear tangent steering problem data

| | |
|---|---|
| Variables | $5N+1$ |
| Constraints | $4(N-1)+7$ |
| Bounds | $N+1$ |
| Linear equality constraints | 7 |
| Linear inequality constraints | 0 |
| Nonlinear equality constraints | $4(N-1)$ |
| Nonlinear inequality constraints | 0 |
| Nonzeros in $\nabla^2 f(x)$ | 0 |
| Nonzeros in $c'(x)$ | $20(N-1)+7$ |

**Performance**

We provide results with the AMPL formulation on a Sun UltraSPARC2. This problem has also been coded in C and solved using SNOPT, both with hand-coded gradients and Jacobians and with ADIC generated gradients and Jacobians. Plots of the position, velocity, and control variables are shown in Figure 11.1.

All of the solvers were given the same initial values of straight lines between the boundary conditions, except for the control $u$ and the first position coordinate $y_1$. The starting value for the control was set to a straight line between $-1$ and $+1$, while the first position coordinate was set to a straight line between 0 and $+1$. The initial value for the final time was $t_f = 1$.

Table 11.2 shows the computational results from AMPL for various values of $N$. Note that LOQO and MINOS fail to solve this problem.

Table 11.2: Performance of AMPL solvers

| Solver | $N = 10$ | $N = 50$ | $N = 100$ | $N = 500$ |
|---|---|---|---|---|
| LANCELOT | Yes | Yes | Yes | Yes |
| $t_f$ | 0.5575747859 | 0.5546725422 | 0.5545925691 | 0.5545368572 |
| iters \| sec | 166 \| 0.90 | 268 \| 26.17 | 419 \| 180.75 | 786 \| 2100.54 |
| MINOS | Yes | infeasible | Yes | infeasible |
| $t_f$ | 0.5575751656 | - | 0.5545958978 | - |
| iters \| sec | 120 \| 0.15 | 1311 \| 1.81 | 923 \| 7.99 | 2933 \| 42.08 |
| SNOPT | Yes | Yes | Yes | Yes |
| $t_f$ | 0.5575751655 | 0.5546728269 | 0.5545959338 | 0.554572935 |
| iters \| sec | 218 \| 0.67 | 414 \| 3.99 | 708 \| 20.96 | 3755 \| 980.40 |
| LOQO | Yes | iteration limit | iteration limit | - |
| $t_f$ | 0.5575751656 | 0.6471450279 | 0.5950385081 | - |
| iters \| sec | 337 \| 3.88 | 10000 \| 242.39 | 10000 \| 704.53 | - |

Table 11.3 shows the computational results for the hand-coded and ADIC augmented C implementations for various values of $N$. As you can see, the ADIC version is considerably slower than the hand-coded version, with the constraint/Jacobian function being about 27 times slower. However, in comparison to the AMPL version, the ADIC version is only about 2.75 times slower for the whole computation.

Table 11.3: Performance of SNOPT with C implementation

| Solver | $N = 10$ | $N = 50$ | $N = 100$ | $N = 500$ |
|---|---|---|---|---|
| SNOPT (hand) | Yes | Yes | Yes | Yes |
| $t_f$ | 0.5575751655263 | 0.55467279242138 | 0.55459588591195 | 0.55457186867 |
| iters | 189 | 448 | 847 | 6157 |
| constraint (sec) | 0.05 | 0.15 | 0.53 | 2.33 |
| objective (sec) | 0.01 | 0.01 | 0.03 | 0.13 |
| solve (sec) | 0.38 | 2.86 | 23.53 | 721.33 |
| SNOPT (ADIC) | Yes | Yes | Yes | Yes |
| $t_f$ | 0.55757516552631 | 0.55467279242138 | 0.55459588591195 | 0.55457186867323 |
| iter | 189 | 448 | 847 | 6157 |
| constraint(sec) | 1.00 | 3.51 | 20.96 | 237.75 |
| objective (sec) | 0.01 | 0.02 | 0.05 | 0.14 |
| solve (sec) | 1.21 | 7.24 | 51.10 | 940.12 |

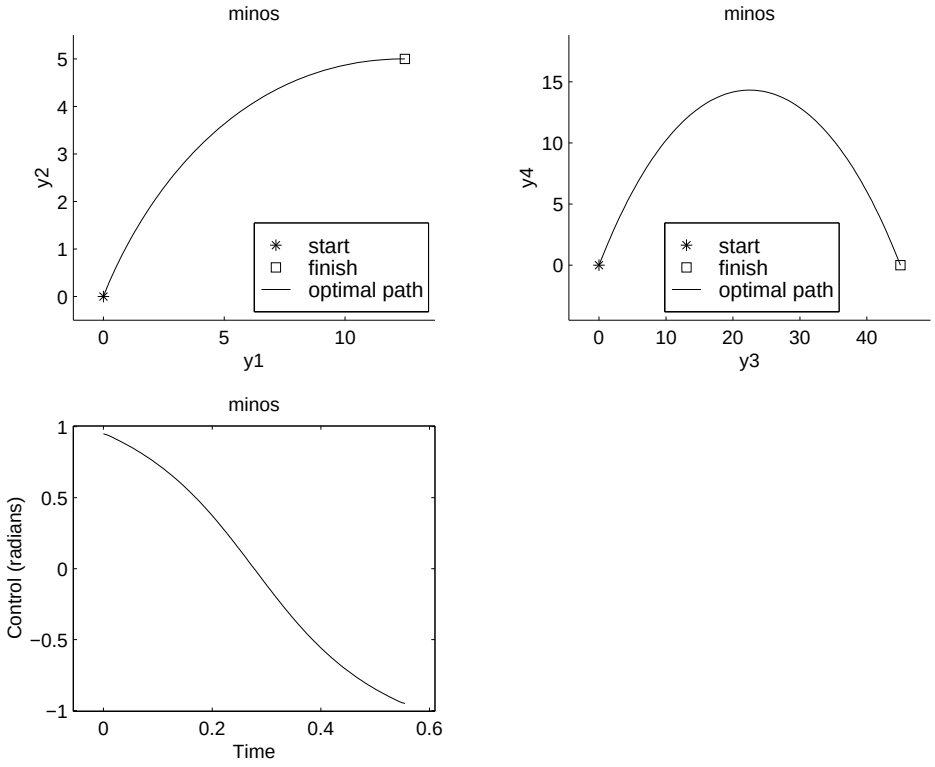


Figure 11.1: Linear Tangent Steering Problem Optimal Path

## 12 Goddard Rocket (Betts, Eldersveld, Huffman [4])

Maximize the final velocity of a vertically launched rocket, using the thrust as a control and subject to boundary conditions, path constraints and physical laws. The rocket is a single-stage vehicle with a finite amount of propellant. Solving this problem should describe an optimal program for the thrust, so as to maximize the final velocity.

### Formulation

The equations of motion for a point mass acted upon by a thrust force of magnitude $T$ are

$$\dot{h} = v, \qquad \dot{v} = \frac{T - D(h,v)}{m} - g, \qquad \dot{m} = -\frac{T}{c}. \tag{12.1}$$

where

$$\begin{array}{rcl} h(t) & \equiv & \text{altitude} \\ v(t) & \equiv & \text{vertical velocity} \\ m(t) & \equiv & \text{rocket mass} \\ T(t) & \equiv & \text{thrust magnitude} \\ t_f & \equiv & \text{final time} \end{array}$$

The function $D$ and the various parameters in (12.1) are

$$D(h,v) = D_0 v^2 \left( e^{-\frac{h}{h_r}} \right), \qquad D_0 = 0.711 \frac{T_M}{c^2},$$

$$T_M = 2 m_0 g, \quad m_0 = 3, \quad g = 32.174, \quad h_r = 23800, \quad c^2 = 3.264 g h_r$$

where $g$ is gravity, $T_M$ is the maximum thrust possible with the rocket engine and $m_0$ is the initial mass of the rocket. The bounds on the state variables are

$$m(t) \geq 1, \qquad t_f \geq 0, \qquad 0 \leq T(t) \leq T_M.$$

The constraints are (12.1), and the boundary conditions, as given in [4], are

$$h(0) = 0, \qquad v(0) = 0, \qquad m(0) = 3, \qquad m(t_f) = 1$$

Discretization of the problem was done using a uniform time step and the trapezoidal rule for the integration of the system over $N$ points. The variables of the optimization problem are

$$h(1:N), \quad v(1:N), \quad m(1:N), \quad T(1:N), \quad t_f.$$

Data for this problem is shown in Table 12.1.

### Performance

We provide results with the AMPL formulation on a Sun UltraSPARC2. All solvers were given the same initial values of straight lines between the boundary conditions for the mass $m$. The initial values for the altitude and the velocity were straight lines between 0 and

Table 12.1: Goddard rocket problem data

| | |
|---|---|
| Variables | $4N+1$ |
| Constraints | $3(N-1)+4$ |
| Bounds | $2N+1$ |
| Linear equality constraints | 4 |
| Linear inequality constraints | 0 |
| Nonlinear equality constraints | $3(N-1)$ |
| Nonlinear inequality constraints | 0 |
| Nonzeros in $\nabla^2 f(x)$ | 0 |
| Nonzeros in $c'(x)$ | $19(N-1)+4$ |

1000 and between 0 and 100 respectively. The initial value for the thrust $T$ was a constant thrust of $T_M/2$.

Table 12.2 shows the computational results for various values of $N$. It is interesting to note that MINOS seems to be the only solver that can solve this problem. Figure 12.1 has plots of the solutions for altitude, velocity, mass, and thrust versus time, as solved by MINOS at $N = 100$.

Table 12.2: Performance of AMPL solvers

| Solver | $N = 10$ | $N = 50$ | $N = 100$ | $N = 500$ |
|---|---|---|---|---|
| LANCELOT | too many iterations | too many iterations | too many iterations | - |
| $v_f$ | 1503.645929 | -0.0004547359203 | -40.99063733 | - |
| iters \| sec | 1000 \| 11.08 | 1000 \| 26.12 | 1000 \| 320.13 | - |
| MINOS | Yes | Yes | Yes | Yes |
| $v_f$ | 1062.028455 | 1060.357748 | 1060.313388 | 1009.468519 |
| iters \| sec | 95 \| 0.12 | 737 \| 2.97 | 2518 \| 18.42 | 2758 \| 73.44 |
| SNOPT | too many iterations | too many iterations | too many iterations | - |
| $v_f$ | 22453.37014 | 7357.058908 | 1244.953645 | - |
| iters \| sec | 9103 \| 10.29 | 136066 \| 444.06 | 166001 \| 1263.53 | - |
| LOQO | infeasible | iteration limit | iteration limit | - |
| $v_f$ | - | 609.1607518 | -162702.75 | - |
| iters \| sec | 1036 \| 17.15 | 5000 \| 147.5 | 5000 \| 517.52 | - |

Table 12.3 shows the computational results for SNOPT solving the Goddard problem for each of the usual $N$. In this case, comparing the results to the AMPL version is not useful, since the AMPL version uses an older version of SNOPT which was unable to solve this problem.

Table 12.3: Performance of SNOPT with C implementation

| Solver | $N = 10$ | $N = 50$ | $N = 100$ |
|---|---|---|---|
| SNOPT | Yes | Yes | Yes |
| $v_f$ | 1033.2418134500 | 1032.8962915064 | 1032.9153331908 |
| iters | 536 | 1803 | 6513 |
| constraint (sec) | 2.29 | 15.56 | 13.13 |
| objective (sec) | 0.04 | 0.13 | 0.10 |
| solve (sec) | 2.92 | 18.06 | 23.51 |

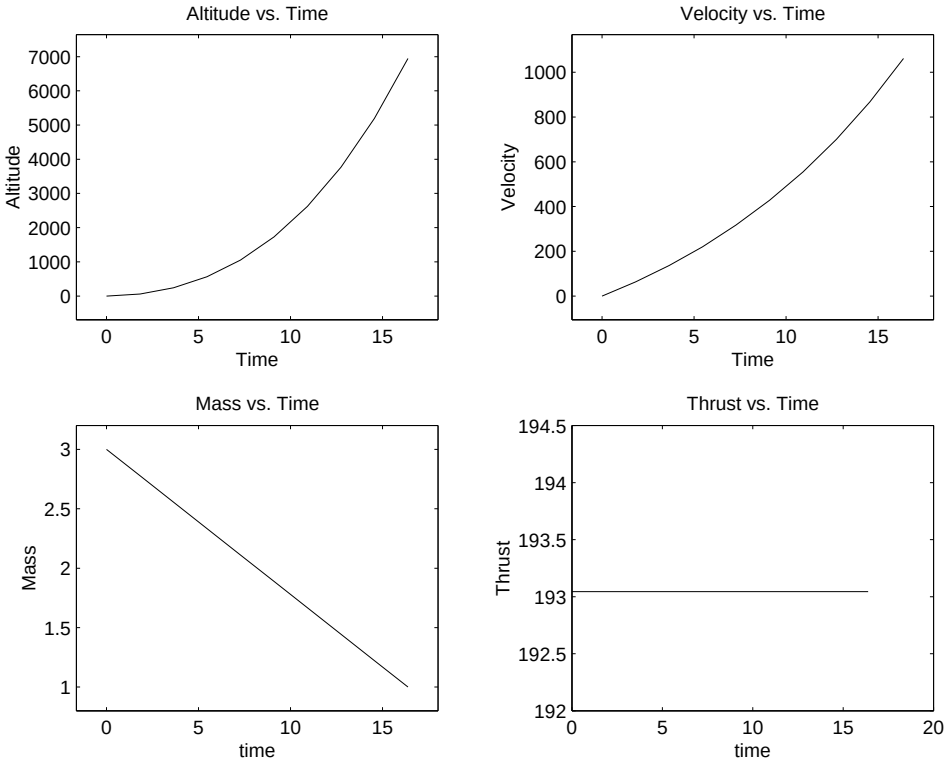


Figure 12.1: Goddard Rocket Problem

## 13 Hang Glider (Betts, Eldersveld, Huffman [4])

Maximize the final horizontal position of a hang glider while satisfying boundary conditions, path constraints, and physical laws. This problem describes the optimal control of a hang glider in the presence of a specified thermal updraft. The objective is to fly the glider as far in the horizontal direction as is possible within a fixed amount of time.

### Formulation

The planar equations of motion for the hang glider are

$$\ddot{x} = \frac{1}{m}(-L\sin(\eta) - D\cos(\eta)), \qquad \ddot{y} = \frac{1}{m}(L\cos(\eta) - D\sin(\eta) - W) \tag{13.1}$$

where $W = mg$ and

$$\begin{array}{rcl} x(t) & \equiv & \text{horizontal position} \\ y(t) & \equiv & \text{altitude} \\ v_x(t) & \equiv & \text{horizontal velocity} \\ v_y(t) & \equiv & \text{vertical velocity} \\ c_L(t) & \equiv & \text{aerodynamic lift coefficient} \end{array}$$

The functions $\eta$, $D$, and $L$ depend on $x$, $v_x = \dot{x}$, $v_y = \dot{y}$, and the control function $c_L$. The function $\eta$ is defined by

$$\sin(\eta) = \frac{V_y(x, v_y)}{v_r(x, v_x, v_y)}, \qquad \cos(\eta) = \frac{v_x}{v_r(x, v_x, v_y)}, \qquad v_r(x, v_x, v_y) = \sqrt{v_x^2 + V_y(x, v_y)^2}$$

where

$$V_y(x, v_y) = v_y - u_a(x), \qquad u_a(x) = u_M(1 - X(x))e^{-X(x)}, \qquad X(x) = (\frac{x}{R} - 2.5)^2,$$

and constants $u_M = 2.5$ and $R = 100$. The functions $D$ and $L$ are defined by

$$D(x, v_x, v_y, c_L) = \frac{1}{2}\left(c_0 + kc_L^2\right)\rho S v_r(x, v_x, v_y)^2, \qquad L(x, v_x, v_y, c_L) = \frac{1}{2}c_L\rho S v_r(x, v_x, v_y)^2,$$

where

$$c_0 = 0.034, \quad k = 0.069662, \qquad m = 100, \quad S = 14, \quad \rho = 1.13, \quad g = 9.80665$$

The only bound is on the control function $c_L$

$$0 \le c_L \le 1.4,$$

The constraints are the system of differential equations (13.1), and the boundary conditions, as given in [6], are

$$x(0) = 0, \qquad y(0) = 1000, \qquad y(t_f) = 900,$$
$$v_x(0) = v_x(t_f) = 13.227567500, \qquad v_y(0) = v_y(t_f) = -1.2876005200.$$

Implementation of the problem involved using a uniform time step and trapezoidal rule for the integration of the system over $N$ grid points. In [4], the final time is left as a user defined parameter. In this implementation $t_f = 100$, since this makes a comparison possible with the results from [6]. An optimization problem is obtained by using the final horizontal position $x(t_f)$ as the merit function to be maximized, and the discretization of (13.1) as the constraints. This formulation leads to an optimization problem with variables

$$x(1:N), \quad y(1:N), \quad v_x(1:N), \quad v_y(1:N), \quad c_L(1:N)$$

Data for this problem is shown in Table 13.1.

Table 13.1: Hang glider problem data

| | |
|---|---|
| Variables | $5N$ |
| Constraints | $4(N-1)+7$ |
| Bounds | $N$ |
| Linear equality constraints | 7 |
| Linear inequality constraints | 0 |
| Nonlinear equality constraints | $4(N-1)$ |
| Nonlinear inequality constraints | 0 |
| Nonzeros in $\nabla^2 f(x)$ | 0 |
| Nonzeros in $c'(x)$ | $24(N-1)+7$ |

**Performance**

We provide results with the AMPL formulation on a Sun UltraSPARC2. All solvers were given the same initial values. For the horizontal position $x$, the initial value is a straight line between 0 and 100. For $(y, v_x, v_y)$, the initial values are straight lines between the boundary conditions. Lastly, for the control $c_L$ a constant initial value of 0.7 was given to the solvers.

Table 13.2: Performance of AMPL solvers

| Solver | $N = 10$ | $N = 50$ | $N = 100$ | $N = 500$ |
|---|---|---|---|---|
| LANCELOT | Yes | Yes | too many iterations | - |
| $x_f$ | 1698.331288 | 1281.02131 | 81.86073975 | - |
| iters \| sec | 117 \| 1.61 | 163 \| 29.13 | 1000 \| 596.19 | - |
| MINOS | infeasible | infeasible | unbounded | unbounded |
| $x_f$ | - | - | - | - |
| iters \| sec | 3825 \| 3.38 | 1716 \| 3.05 | 2232 \| 18.76 | 3942 \| 117.88 |
| SNOPT | Yes | Yes | Yes | Yes |
| $x_f$ | 1716.750091 | 1055.075921 | 1255.190371 | 1247.405707 |
| iters \| sec | 229 \| 0.49 | 1872 \| 14.96 | 3622 \| 66.74 | 26651 \| 1901.27 |
| LOQO | iteration limit | iteration limit | iteration limit | - |
| $x_f$ | 1143.042306 | 162.5977425 | 442.7979116 | - |
| iters \| sec | 10000 \| 11.34 | 10000 \| 26.42 | 10000 \| 73.16 | - |

Table 13.2 presents computational results for various values of $N$. SNOPT found a solution which for the largest $N$, is identical to the solution described in [6]. LOQO found

nearly the correct solution for the $x$ and $y$ states, but was wildly off for the rest of the variables. MINOS was not able to solve the system for any problem size.

Table 13.3: Performance of SNOPT with C implementation

| Solver | $N = 10$ | $N = 50$ | $N = 100$ | $N = 500$ |
|---|---|---|---|---|
| SNOPT | Yes | Yes | Yes | Yes |
| $x_f$ | 1889.2567964520 | 1285.5323615 | 1255.2241243 | 1247.2051276964 |
| iters | 585 | 1923 | 4509 | 43832 |
| constraint (sec) | 0.30 | 0.80 | 1.69 | 23.41 |
| objective (sec) | 0.01 | 0.02 | 0.03 | 0.42 |
| solve (sec) | 0.91 | 6.89 | 29.69 | 2036.29 |

Table 13.3 shows the computational results generated by the C code which calls SNOPT for the various values of $N$. Note that the C code is faster for the $N$. However, for $N = 500$ the AMPL version is faster and takes fewer iterations. This is perhaps due to AMPL.

Figure 13.1 shows plots for $N = 500$ for each of the variables. These graphs are generated by the AMPL implementation, using SNOPT, but the C coded version generated identical graphs.

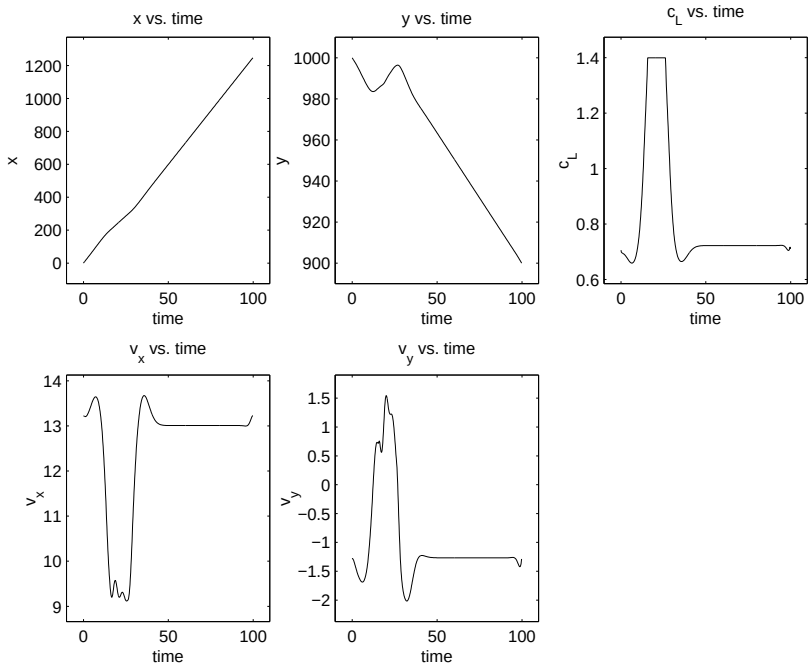


Figure 13.1: Hang Glider Problem

## 14 Implementation of COPS in C

We use the formulation of general constrained optimization problem defined by a merit function $f : \mathbb{R}^n \mapsto \mathbb{R}$ and nonlinear constraints $c : \mathbb{R}^n \mapsto \mathbb{R}^m$:

$$\min \{f(x) : x_l \leq x \leq x_u,\ c_l \leq c(x) \leq c_u\}.$$

We specify the problem by the following functions in C:

- `int name_xb (par_type par, var_type *xl, var_type *xu)`
  specifies the bounds $x_l$ and $x_u$,
- `int name_cb (par_type par, double *cl, double *cu)`
  specifies the bounds $c_l$ and $c_u$,
- `int name_xs (par_type par, var_type *x)`
  specifies the standard starting point,
- `int name_f (par_type par, var_type *x, obj_type obj)`
  specifies the values $f(x)$ and $\nabla f(x)$,
- `int name_c (par_type par, var_type *x, con_type con)`
  specifies the values $c(x)$ and $c'(x)$.
- `int name_sp (par_type par, int *nnz, int *ipntr, int *indcol)`
  specifies the sparsity pattern of the sparse Jacobian $c'(x)$

where `name` is the name of the problem (e.g. polygon, electrns, etc.). Here `obj_type` and `con_type` are objective and constraint types defined as follows:

```
typedef struct {
  double *f;          /* pointer to the objective value   */
  double *grad;       /* array of the partial derivatives */
} obj_type;

typedef struct {
  double *c;          /* array of constraints (of length m)                */
  int    *nnz;        /* Jacobian - pointer to number of nonzeros          */
  int    *ipntr;      /* Jacobian - row "pointers" (array of length m+1)   */
  int    *indcol;     /* Jacobian - column indicies (array of length *nnz) */
  double *jacrow;     /* Jacobian - nonzero entries (array of length *nnz) */
} con_type;
```

and `par_type` and `var_type` are problem dependent parameter and variable types, respectively. We use the compressed sparse row storage for the Jacobian, but we provide a routine `row2col` that changes from compressed sparse row storage to compressed sparse column storage, used by some solvers in Fortran 77.

We decided to combine both linear and nonlinear parts of the Jacobian $c'(x)$ in `name_c.c`. However, it is still possible to separate them for such solvers as SNOPT if there is a significant number of linear constraints. In this case the user would have to reorder the constraints in some cases.

### 14.1 Largest Small Polygon

```
typedef struct {
  double r;           /* polar radius from a fixed vertex */
  double theta;       /* polar angle from a fixed vertex  */
} var_type;

typedef int par_type; /* number of vertices in a polygon  */
```

### 14.2 Electrons on a Sphere

```
typedef struct {
  double x;         /* x-coordinate of a point charge */
  double y;         /* y-coordinate of a point charge */
  double z;         /* z-coordinate of a point charge */
} var_type;

typedef int par_type;     /* number of point charges  */
```

### 14.3 Saw Path Tracking

```
typedef double var_type;       /* polynomial coefficients */

typedef struct{
  int     d;          /* maximum degree of the polynomial */
  int     N;          /* number of data points            */
  double *x;          /* array of x-values of data points */
  double *y;          /* array of y-values of data points */
  double  M;          /* initial slope of the polynomial  */
  double  R;          /* minimum radius of curvature      */
} par_type;
```

### 14.4 Hanging Chain

```
typedef double var_type;  /* height of the chain from a fixed horizontal */

typedef struct {
  int    nh;             /* number of discretization points       */
  double L;              /* length of the chain                   */
  double a;              /* height of the chain on the left side  */
  double b;              /* height of the chain on the right side */
} par_type;
```

### 14.5 Optimal Shape Design of a Cam

```
typedef double var_type;     /* polar radius of the edge points of the cam */

typedef struct{
  int n;                        /* number of points in the discretization */
  double R_min;                 /* minimal allowed radius                 */
  double R_max;                 /* maximal allowed radius                 */
  double R_v  ;                 /* valve parameter                        */
  double alpha;                 /* curvature parameter                    */
  double d_theta;               /* change in angle = 2*pi/5/(n+1)         */
} par_type;
```

### 14.6 Isometrization of Alpha-Pinene

```
typedef struct {
  double v;            /* parameters determining piecewise polynomial on the  */
  double w[4];         /*    interval to the right of the grid point          */
} grid_type;

typedef struct {
  double     theta[5]; /* reaction coefficients                                */
  grid_type *u[5];     /* pointers to the piecewise polynomial representation */
} var_type;            /*    of the chemical components quantities components */

typedef struct {
  int     nh;          /* number of grid points in the uniform partitioning   */
  int     nm;          /* number of concentration measurements                */
  double  t_f;         /* final time: diff equations are solved on [0,t_f]    */
  double  y_0[5];      /* initial conditions for the differential equations   */
  double *tau;         /* array of times of the concentration measurements    */
  double *z[5];        /* arrays of the concentration measurements of the     */
} par_type;            /*    five chemical components in the reaction         */
```

### 14.7 Marine Population Dynamics

```
typedef struct {
  double v;            /* parameters determining piecewise polynomial on the  */
  double w[4];         /*    interval to the right of the grid point          */
} grid_type;

typedef struct {
  double     m[MAXNS]; /* mortality coefficients for the stage i              */
  double     g[MAXNS-1];  /* growth coefficients from stage i to stage i+1    */
  grid_type *u[MAXNS]; /* pointers to the piecewise polynomial representation */
} var_type;            /*    of the population stage abundances               */
```

```
typedef struct {
  int     nh;          /* number of grid points in the uniform partitioning   */
  int     ns;          /* number of stages in the population                  */
  int     nm;          /* number of population stage abundance measurements   */
  double  t_f;         /* final time: diff. equations are solved on [0,t_f]   */
  double *tau;         /* array of times of the stage abundance measurements  */
  double *z[MAXNS];    /* arrays of the stage abundance measurements          */
} par_type;
```

### 14.8 Flow in a Channel

```
typedef struct {
  double v[4];         /* parameters determining piecewise polynomial on the  */
  double w[4];         /*    interval to the right of the grid point          */
} var_type;

typedef struct {
  int     nh;          /* number of grid points in the uniform partitioning   */
  double  R;           /* Reinolds number                                     */
  double  u_0[2];      /* boundary conditions for the differential equation   */
  double  u_1[2];      /*    at t=0 and t=1                                   */
} par_type;
```

### 14.9 Non-inertial Robot Arm

```
typedef struct {
  double rho;           /* length of arm              */
  double the;           /* theta angle for arm        */
  double phi;           /* phi angle for arm          */
  double rho_dot;       /* rho velocity               */
  double the_dot;       /* theta velocity             */
  double phi_dot;       /* phi velocity               */
  double u_rho;         /* control in rho direction   */
  double u_the;         /* control in theta direction */
  double u_phi;         /* control in phi direction   */
} oth_type;

typedef struct {
  oth_type *vars;           /* stuct of the variables */
  double h;                 /* time step              */
} var_type;
```

### 14.10 Linear Tangent Steering

```
typedef struct {
  double y1;     /* first position coordinate    */
  double y2;     /* second position coordinate   */
  double y3;     /* first velocity coordinate    */
  double y4;     /* second velocity coordinate   */
  double u;      /* control coordinate (radians) */
} oth_type;

typedef struct {
  oth_type *vars;      /* stuct of the variables */
  double h;            /* time step              */
} var_type;

typedef int par_type;  /* number of grid points */
```

### 14.11 Goddard Rocket

```
typedef struct {
  double h;           /* altitude           */
  double v;           /* vertical velocity  */
  double m;           /* mass               */
  double T;           /* Thrust             */
} oth_type;

typedef struct {
  oth_type *vars;     /* stuct of the variables */
  double h;           /* time step              */
} var_type;

typedef int par_type;  /* number of grid points */
```

### 14.12 Hang Glider

```
typedef struct {
  double x;      /* first position coordinate    */
  double y;      /* second position coordinate   */
  double vx;     /* first velocity coordinate    */
  double vy;     /* second velocity coordinate   */
  double cL;     /* control coordinate (radians) */
} oth_type;

typedef struct {
  oth_type *vars;      /* stuct of the variables */
} var_type;

typedef int par_type;  /* number of grid points  */
```

## References


[1] M. Anitescu and R. Serban, *A sparse superlinearly convergent SQP with applications to two-dimensional shape optimization*, Preprint ANL/MCS-P706-0198, Argonne National Laboratory, Argonne, Illinois, 1998.

[2] U. M. Ascher, R. M. M. Mattheij, and R. D. Russell, *Numerical solution of boundary value problems for ordinary differential equations*, SIAM, 1995.

[3] B. M. Averick, R. G. Carter, J. J. Moré, and G.-L. Xue, *The MINPACK-2 test problem collection*, Preprint MCS-P153-0694, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Illinois, 1992.

[4] J. Betts, S. Eldersveld, and W. Huffman, *Sparse nonlinear programming test problems (Release 1.0)*, Technical report BCSTECH-93-047, Boeing Computer Services, Seattle, Washington, 1993.

[5] G. E. P. Box, W. G. Hunter, J. F. MacGregor, and J. Erjavec, *Some problems associated with the analysis of multiresponse data*, Technometrics, 15 (1973), pp. 33–51.

[6] R. Bulirsch, E. Nerz, H. J. Pesch, , and O. von Stryk, *Combining direct and indirect methods in nonlinear optimal control: Range maximization of a hang glider*, Technical report 313, Technische Universitdt M|nchen, 1993.

[7] T. Erber and G. M. Hockney, *Comment on Method of Constrained Global Optimization*, Phys. Rev. Lett., 74 (1995), p. 1482.

[8] D. Gay, *AMPL models*. `http://www.netlib.org/ampl/models/`.

[9] R. L. Graham, *The largest small hexagon*, J. Combin. Th., 18 (1975), pp. 165–170.

[10] J. R. Morris, D. M. Deaven, and K. M. Ho, *Genetic algorithm energy minimization for point charges on a sphere*, Phys. Rev. B, 53 (1996), pp. R1740–R1743.

[11] B. J. Rothschild, A. F. Sharov, A. J. Kearsley, and A. S. Bondarenko, *Estimating growth and mortality in stage-structured populations*, Journal of Plankton Research, 19 (1997), pp. 1913–1928.

[12] I.-B. Tjoa and L. T. Biegler, *Simultaneous solution and optimization strategies for parameter estimation of differential-algebraic equations systems*, Ind. Eng. Chem. Res., 30 (1991), pp. 376–385.

[13] R. Vanderbei, *Nonliner optimization models*. `http://www.sor.princeton.edu/~rvdb/ampl/nlmodels/`.

[14] R. J. Vanderbei, *LOQO user's manual - Version 3.10*, Technical Report SOR-97-08, Princeton University, Princeton, New Jersey, 1997.

[15] R. J. Vanderbei and D. F. Shanno, *An interior-point algorithm for nonconvex nonlinear programming*, Technical Report SOR-97-21, Princeton University, 1997. To appear in Comp. Optim. Appl.