Arc Search Methods for Linearly Constrained Optimization

Nick Henderson

Institute for Computational and Mathematical Engineering
Stanford University

INFORMS 2014 Annual Meeting

Optimization

Unconstrained optimization:

$$\underset{x}{\text{minimize}} \quad F(x)$$

Linearly constrained optimization:

$$\begin{aligned} \underset{x}{\text{minimize}} &\quad F(x) \\ \text{subject to} &\quad Ax \geq b \end{aligned}$$

Specifics:

- A is a real matrix with m rows and n columns
- b is a real vector of length m
- x_k is always feasible
- have access to first and second derivatives

$$g_k = \nabla F(x_k), \quad H_k = \nabla^2 F(x_k)$$

This talk

Discuss optimization algorithms that:

- are based on Newton's method for fast convergence
- converge to points satisfying the second order necessary optimality conditions
- use second derivatives while avoiding difficult scaling choices
- work on problems with many variables

Present a research implementation:

- showing that the features listed above can be obtained in practice
- compare to `IPOPT` on continuous formulation of the Hamiltonian Cycle Problem

Unconstrained optimization

Generate a sequence of iterates $\{x_0, x_1, x_2, \ldots\}$ that converges to an optimal point x^*.

Major steps at iterate x_k:

1. compute local information
   - $g_k = \nabla F(x_k)$
   - $H_k = \nabla^2 F(x_k)$
2. check termination conditions
   - First order necessary: $g^* = 0$
   - Second order necessary: $H^* \succeq 0$
   - Second order sufficient: $H^* \succ 0$
3. compute new iterate x_{k+1}
   - the secret sauce

Newton's method

The gold standard in optimization is Newton's method:

$$x_{k+1} = x_k - H_k^{-1} g_k$$

The good:

- if $H_k \succ 0$, then x_{k+1} is the minimizer of a quadratic model of F around x_k
- has a quadratic rate of convergence if $H^* \succ 0$ and x_k is close enough to x^*

The bad:

- x_{k+1} is not defined if H_k is singular

The ugly:

- given arbitrary starting point $\{x_k\}$ may diverge, cycle, or converge to maximizer

Basic strategy: descent methods

Descent methods are designed to emulate Newton's method when it works, but enforce convergence to satisfactory points. These methods satisfy the descent property:

$$F(x_{k+1}) < F(x_k) \text{ for } k = 0, 1, 2, \ldots$$

Two common algorithm classes:

- line search methods
- trust region methods

This talk:

- arc search methods

Line search

The process:

1. compute a search direction p_k
2. invoke a line search procedure to compute step size α_k, which ensures $F(x_k + \alpha_k p_k) < F(x_k)$
3. Update: $x_{k+1} = x_k + \alpha_k p_k$

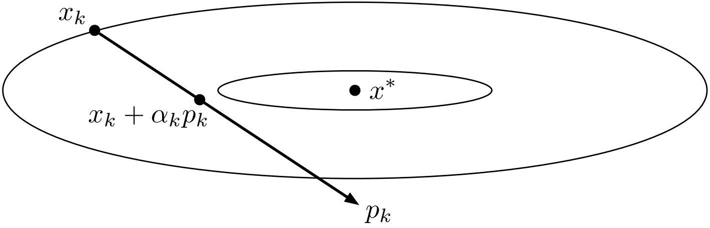
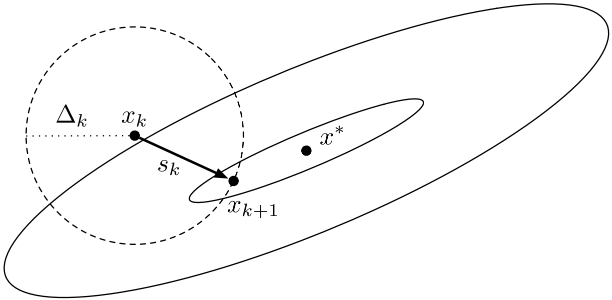


Figure: line search diagram

Trust region

The process:

1. decide trust region radius, Δ_k
2. compute step s_k such that $s_k^T s_k \leq \Delta_k$
3. if $F(x_k + s_k) < F(x_k)$ perform update $x_{k+1} = x_k + s_k$, otherwise decrease Δ_k and recompute s_k



Ok, what's the problem?

Line search:

- must come up with special methods to deal with indefinite H_k, not clear what's best
- not clear how to use directions of negative curvature

Trust region:

- may need to solve linear system multiple times
- scaling of the trust region is an issue
- added subproblem difficulty in presence of constraints

Arc search

Arc search methods construct an arc, Γ_k, at each iteration. A univariate search procedure selects an appropriate step parameter α_k, which gives the update:

$$x_{k+1} = x_k + \Gamma_k(\alpha_k)$$

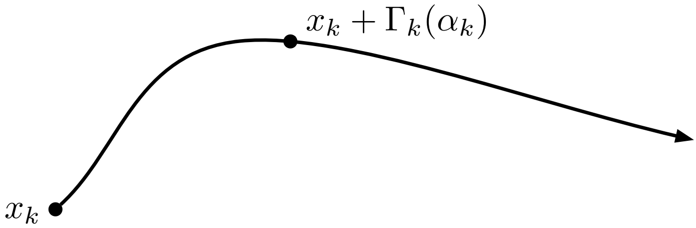


Figure: arc search diagram

Arc search

Arc search methods construct an arc, Γ_k, at each iteration. A univariate search procedure selects an appropriate step parameter α_k, which gives the update:

$$x_{k+1} = x_k + \Gamma_k(\alpha_k)$$

Basic properties:

- $\Gamma_k(\alpha)$ is twice continuously differentiable for $\alpha \in [0, \infty)$
- $\Gamma_k(0) = 0$
- $g_k^T \Gamma_k'(0) < 0$

Issues:

- convergence theory
- definition of arcs

Arc search

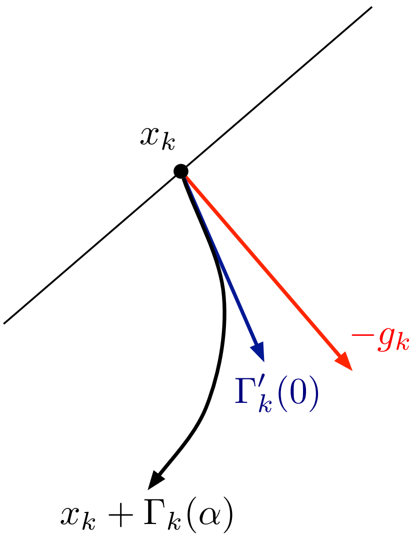


Figure: diagram of basic arc properties

Arc search: $H_k \succ 0$

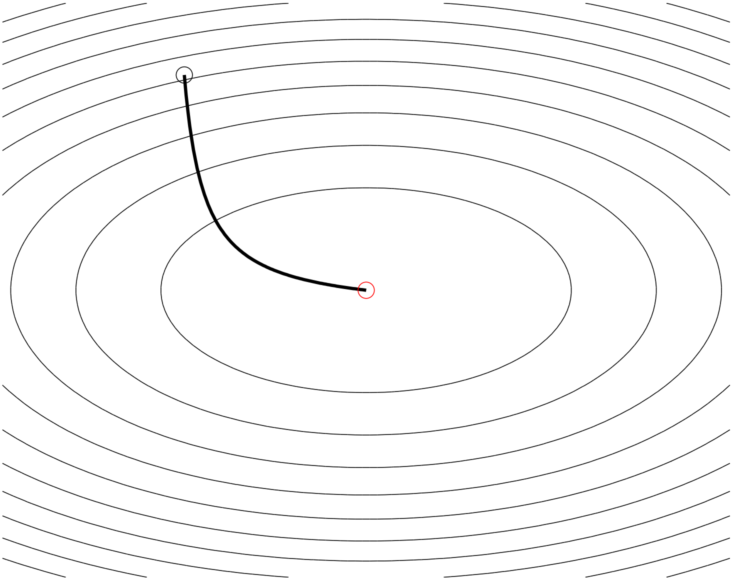


Figure: example arc on positive definite quadratic

Arc search: $H_k \not\succ 0$

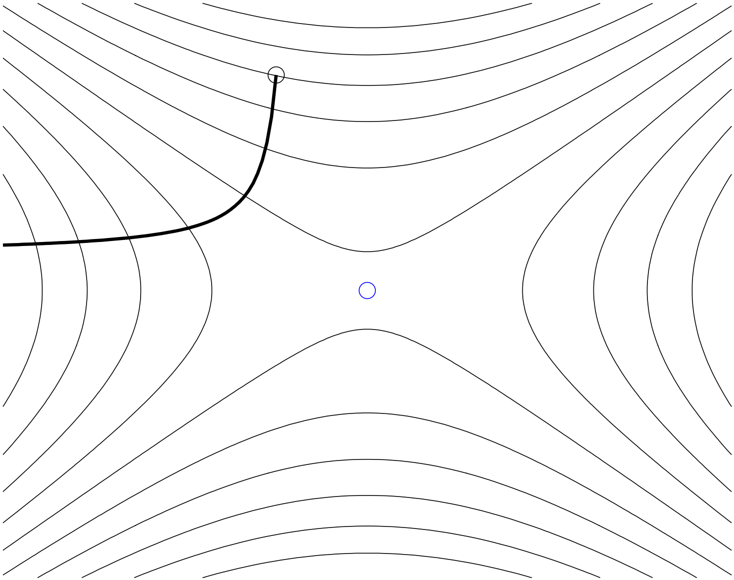


Figure: example arc on indefinite quadratic

Example arcs

Several arcs have been studied. In the following table, s_k is a descent direction and d_k is a direction of negative curvature.

algorithm	$\Gamma_k(\alpha)$
Moré & Sorensen	$\alpha^2 s_k + \alpha d_k$
Goldfarb	$\alpha s_k + \alpha^2 d_k$
Forsgren & Murray	$\alpha(s_k + d_k)$
Behrman	$w(\alpha) : w'(\alpha) = -H_k w(\alpha) - g_k$
Del Gatto	$Qw(\alpha) : w'(\alpha) = -Q^T H_k Q w(\alpha) - Q^T g_k$
`arcopt`	$Qw(\alpha) : (Q^T H_k Q + \pi(\alpha) I) w(\alpha) = -Q^T g_k$

We can prove convergence to second order points with these arcs. For the first three, it is not clear how to scale d_k and we still have to modify H_k to get s_k.

Trust region arc

The trust region subproblem is:

$$\begin{array}{ll} \underset{w}{\text{minimize}} & \frac{1}{2} w^T H_k w + g_k^T w \\ \text{subject to} & \frac{1}{2} w^T w \leq \Delta^2 \end{array}$$

If we have the Lagrange multiplier, π, we can get the solution by solving

$$(H_k + \pi I) w = -g_k$$

We can also think of the solution as an arc parameterized by Δ or π.

Trust region arc details

Given the spectral decomposition $H_k = U \Lambda U^T$, a reparameterized solution to the trust region subproblem is:

$$
\begin{aligned}
w(\alpha) &= -U q(\alpha, \Lambda) U^T g_k \\
q(\alpha, \lambda) &= \frac{\alpha}{\alpha(\lambda - \lambda_{\min}) + 1}
\end{aligned}
$$

- $w(0) = 0$
- $w'(0) = -g_k$, initially steepest descent
- If $H_k \succ 0$, then $\alpha = 1/\lambda_{\min}$ gives the Newton step.
- If $H_k \not\succeq 0$, then $w(s)$ diverges away from saddle point.
- $d/d\alpha \|w(\alpha)\| > 0$ for all $\alpha \geq 0$
- Can apply a perturbation to get second order convergence
- Maintains important properties in 2D subspaces

Linear equality constraints

$$
\begin{array}{ll}
\underset{x}{\text{minimize}} & F(x) \\
\text{subject to} & Ax = b
\end{array}
$$

Can be solved with line search in the nullspace of A:

- start with a feasible point x_0, such that $Ax_0 = b$
- $x_{k+1} = x_k + \alpha_k p$ is feasible if $p \in \mathbf{null}(A)$
- say $p = Zp_z$, with $AZ = 0$. The columns of Z span $\mathbf{null}(A)$
- compute p_z by solving $Z^T H_k Z p_z = -Z^T g_k$
- terminate when $Zg = 0$ and $Z^T H Z$ positive semi-definite

Linear inequality constraints

$$\begin{array}{ll} \underset{x}{\text{minimize}} & F(x) \\ \text{subject to} & Ax \geq b \end{array}$$

- active set methods solve this problem by identifying a set of constraints (rows of A) that give equality at the solution, $\overset{*}{A}x^* = b^*$
- the remaining constraints are inactive (strictly feasible), $\bar{A}x^* > \bar{b}$
- during the procedure, the algorithm must be able to:
    - add a constraint when one is encountered by the search procedure
    - delete a constraint when doing so allows a decrease in the objective function

Restricted and unrestricted steps

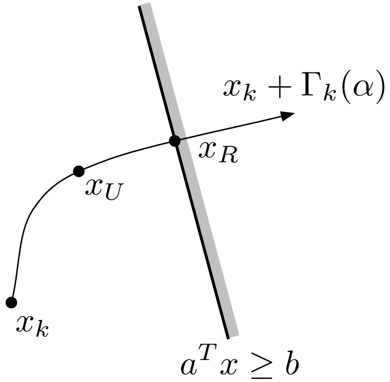


Figure: x_R is a point with a restricted step. x_U is a point with an unrestricted step.

ARCOPT: the implementation

Details:

- a Matlab implementation of a trust region arc search method
- an active set, reduced-gradient method
- sparse and stable factorization and updates with LUSOL
  - allows efficient products with Z
- direction of negative curvature computed with `eigs` on $Z^T HZ$
- modified Newton direction computed with `pcg` or `minres` on $(Z^T HZ + \delta I)p = -Z^T g$
- uses EXPAND procedure to
  - improve conditioning of basis matrix
  - mitigate cycling
  - remove roots at $\Gamma_k(0)$ when deleting constraints

Example application: Hamiltonian cycle problem (HCP)

- a graph is a collection of nodes connected with edges
- a Hamiltonian cycle follows edges to visit each node exactly once and return to the start
- HCP is NP-complete

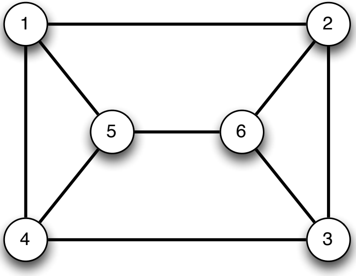


Figure: the envelope graph

Hamiltonian cycle problem (HCP)

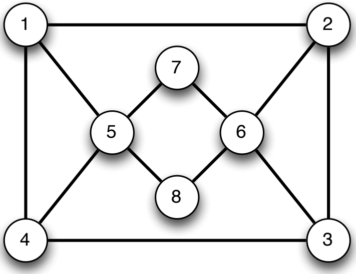


Figure: a graph with no Hamiltonian cycle

Markov chain formulation for HCP

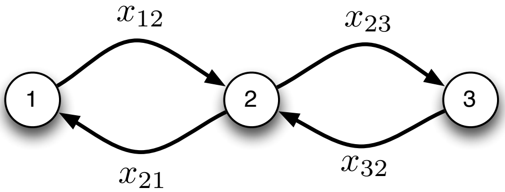


Figure: variables assigned to edges

$$x = \begin{bmatrix} x_{21} \\ x_{12} \\ x_{32} \\ x_{23} \end{bmatrix} \qquad P(x) = \begin{pmatrix} 0 & x_{12} & 0 \\ x_{21} & 0 & x_{23} \\ 0 & x_{32} & 0 \end{pmatrix}$$

Markov chain formulation for HCP

Optimization formulation from Ejov, Filar, Murray, and Nguyen:

$$
\begin{array}{ll}
\underset{x}{\text{minimize}} & G(x) = \det\left(I - P(x) + \frac{1}{N} e e^T\right) \\
\text{subject to} & P(x) e = e \\
 & P(x)^T e = e \\
 & x \geq 0
\end{array}
$$

If the graph has a Hamiltonian cycle, then it can be shown

- $G(x^*) = -N$ is a global minimizer
- elements of x^* are 0 or 1
- 1's correspond to edges that are included in the cycle

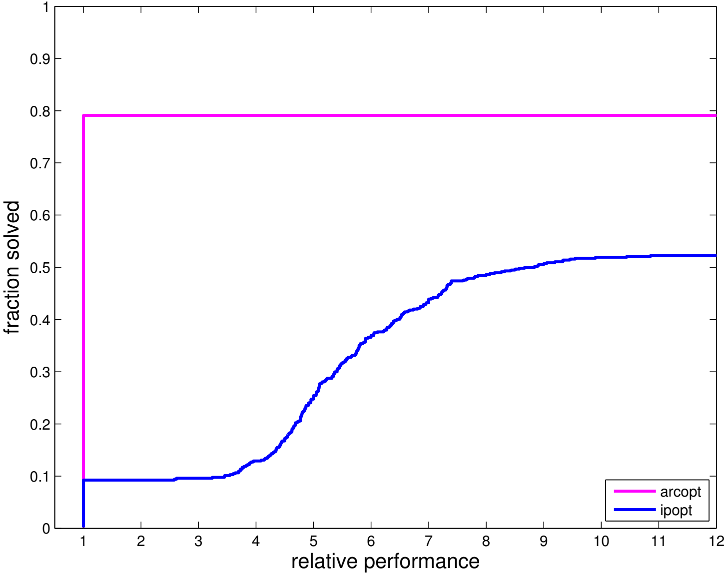
Features:

- it is possible to efficiently compute the first and second derivatives of G
- using directions of negative curvature is particularly important

HCP experiment

- Compare performance of `ARCOPT` with IPOPT on HCP
- IPOPT
  - an interior point code for general nonlinear problems
  - uses second derivatives
- test on all 10, 12, and 14 node cubic graphs with Hamiltonian cycles
  - number of variables is $3\times$ number of nodes
  - number of constraints is $2\times$ number of nodes
- a total of 574 problems (obtained from M. Haythorpe at Flinders University)
- note: SNOPT is not a good candidate for this experiment, because it does not use second derivatives. SNOPT performed poorly in preliminary tests

HCP performance profile (574 graphs)



Thanks!

Vladimir Ejov, Jerzy A. Filar, Walter Murray, and Giang T. Nguyen.
Determinants and longest cycles of graphs.
SIAM Journal on Discrete Mathematics, 22(3):1215–1225, 2008.

Anders Forsgren and Walter Murray.
Newton methods for large-scale linear inequality-constrained minimization.
SIAM Journal on Optimization, 7(1):162–176, 1997.

Jorge J. Moré and Danny C. Sorensen.
On the use of directions of negative curvature in a modified newton method.
Mathematical Programming, 16:1–20, 1979.

B. A. Murtagh and M. A. Saunders.
Large-scale linearly constrained optimization.
Mathematical Programming, 14:41–72, 1978.