The Speed of Shor's R-algorithm

J.V. Burke* A.S. Lewis† M.L. Overton‡

Dedicated to M.J.D. Powell on the Occasion of his 70th Birthday
Submitted May 8, 2007

Key words: Shor's r-algorithm, space dilation, linear convergence, unconstrained optimization, nonsmooth optimization.

AMS 2000 Subject Classification: 90C30, 65K05

Abstract

Shor's r-algorithm is an iterative method for unconstrained optimization, designed for minimizing nonsmooth functions, for which its reported success has been considerable. Although some limited convergence results are known, nothing seems to be known about the algorithm's rate of convergence, even in the smooth case. We study how the method behaves on convex quadratics, proving linear convergence in the two-dimensional case and conjecturing that the algorithm is always linearly convergent, with an asymptotic convergence rate that is independent of the conditioning of the quadratic being minimized.

*Department of Mathematics, University of Washington, Seattle, WA 98195, U.S.A. burke@math.washington.edu. Research supported in part by National Science Foundation Grant DMS-0505712.

†School of Operations Research and Information Engineering, Cornell University, Ithaca, NY 14853, U.S.A. aslewis@orie.cornell.edu. Research supported in part by National Science Foundation Grant DMS-0504032.

‡Courant Institute of Mathematical Sciences, New York University, New York, NY 10012, U.S.A. overton@cs.nyu.edu. Research supported in part by National Science Foundation Grant DMS-0412049.

1 Introduction

Shor's r-algorithm [Sho85, Section 3.6] was designed primarily to minimize nonsmooth functions, something that it does quite effectively according to extensive results of Shor and others, particularly Kappel and Kunsevich [KK00]. The r-algorithm, which can be viewed as a variable metric method that does not satisfy the secant equation, should not be confused with Shor's subgradient method with space dilation [Sho85, Section 3.3], which is related to the symmetric rank-one quasi-Newton method [Tod86]. The r-algorithm also uses space dilation, but in the direction of the difference of two successive gradients (or subgradients, in the nonsmooth case). The r-algorithm's crucial parameter γ (see below) ranges between 0 and 1: when $\gamma = 0$, the method reduces to steepest descent, while when $\gamma = 1$, it reduces to a variant of the conjugate gradient method [Sho85, p.70]. Some limited convergence results are known; a result is given for continuous, piecewise smooth functions in [Sho85, Thm 3.13]. However, nothing seems to be known about the convergence rate of the algorithm, even in the smooth case.

This paper studies the rate of convergence of the r-algorithm on convex quadratics, conjecturing that the method is linearly convergent in this case, with a proof when $n = 2$. We also make the stronger conjecture that, for $\gamma \in (0, 1)$, the algorithm is linearly convergent with a rate that is independent of the conditioning of the quadratic to which it is applied, a result that would interpolate nicely between well known results for steepest descent and conjugate gradient.

2 Shor's r-algorithm

For a smooth function $f : \mathbf{R}^n \to \mathbf{R}$, the algorithm fixes a constant $\gamma \in [0, 1]$, begins with an initial point $x_0 \in \mathbf{R}^n$, defines an initial matrix $B_0 = I$ (the identity matrix), and then iterates as follows, for step $k = 0, 1, 2, \ldots$:

$$\begin{aligned} x_{k+1} &= x_k - t_k B_k B_k^T \nabla f(x_k) \\ &\quad \text{where } t_k \text{ minimizes } f(x_{k+1}) \\ r_{k+1} &= B_k^T(\nabla f(x_{k+1}) - \nabla f(x_k)) \quad \text{normalized} \\ B_{k+1} &= B_k(I - \gamma r_{k+1} r_{k+1}^T). \end{aligned}$$

Here and below, "normalized" means normalized using the 2-norm. In practice, an inexact line search would be used to obtain t_k, but for the purposes

of our analysis, we make the following assumption throughout:

Exact line search: The step size t_k globally minimizes $f(x_{k+1})$.

Notice setting $\gamma = 0$ just gives the method of steepest descent.

We can interpret the algorithm as making steepest descent steps in a transformed space, as follows. At step k, given the current iterate x_k and the current transformation matrix B_k, we first make the change of variables $y = B_k^{-1}x$. In terms of this new variable, we wish to minimize the function

$$h_k(y) = f(x) = f(B_k y).$$

Our current point is $y_{\text{old}} = B_k^{-1}x_k$. Starting from this point, a steepest descent step takes us to the new point

$$y_{\text{new}} = y_{\text{old}} - t_k \nabla h_k(y_k),$$

where t_k minimizes $h_k(y_{\text{new}})$, and from y_{new} we invert the change of variables to obtain x_{k+1}. Hence we deduce the iteration:

$$x_{k+1} = B_k y_{k+1} = B_k\Big(y_k - t_k \nabla h_k(y_k)\Big) = x_k - t_k B_k B_k^T \nabla f(x_k).$$

The update to the transformation matrix B_k is motivated by the next assumption, that again we make throughout:

Convex quadratic: $f(x) = \frac{1}{2}x^T Ax$ for a symmetric positive-definite matrix A.

In this case it is easy to verify the relationship

$$x_1^T\Big(\nabla f(x_1) - \nabla f(x_0)\Big) = 0.$$

In other words, one steepest descent iteration on a convex quadratic results in a difference of successive gradients that is orthogonal to the direction to the minimizer. The transformation represented by the matrix B_k dilates the space in the direction of this gradient difference, thereby encouraging future iterations in orthogonal directions. In particular, in the limiting case $\gamma = 1$, all future iterations must be orthogonal to the gradient difference, resulting in the conjugate gradient iteration. More generally, the method can be viewed as a variable metric method, but one for which the updated matrix does not satisfy the well known secant equation.

The classical theory for the method of steepest descent relates the rate of decrease in the function value to the conditioning of the quadratic. Specifically, we have

$$x_1^T A x_1 \;\leq\; \left(\frac{\kappa(A)-1}{\kappa(A)+1}\right)^2 x_0^T A x_0,$$

where $\kappa(A) = \|A\|\|A^{-1}\|$, the condition number of A [Lue84]. In subsequent iterations, applying the same inequality in the transformed space shows

$$\begin{aligned} x_{k+1}^T A x_{k+1} &= y_{\text{new}}^T B_k^T A B_k y_{\text{new}} \\ &\leq \left(\frac{\kappa(B_k^T A B_k)-1}{\kappa(B_k^T A B_k)+1}\right)^2 y_{\text{old}}^T B_k^T A B_k y_{\text{old}} \\ &= \left(\frac{\kappa(B_k^T A B_k)-1}{\kappa(B_k^T A B_k)+1}\right)^2 x_k^T A x_k, \end{aligned}$$

since y_{new} is obtained from y_{old} by one iteration of steepest descent on the function

$$h_k(y) = \frac{1}{2} y^T B_k^T A B_k y.$$

Consequently, to understand the speed of Shor's r-algorithm in the case of a convex quadratic with an exact line search, we must understand how the condition number of the matrix

$$A_k = B_k^T A B_k$$

evolves as the step counter k grows.

To study the Shor iteration, we make some changes of variables. In our quadratic case, assuming the iteration does not terminate with some $x_k = 0$, the iteration becomes

$$\begin{aligned} t_k &= \frac{x_k^T A B_k B_k^T A x_k}{x_k^T A B_k B_k^T A B_k B_k^T A x_k} \\ x_{k+1} &= x_k - t_k B_k B_k^T A x_k \\ r_{k+1} &= B_k^T A (x_{k+1} - x_k) \;\text{ normalized} \\ B_{k+1} &= B_k (I - \gamma r_{k+1} r_{k+1}^T). \end{aligned}$$

Notice that the iteration is well-defined because $x_k \neq 0$. If we define a new (nonzero) variable

$$z_k = B_k^T A x_k,$$

we obtain

$$\begin{aligned} t_k &= \frac{\|z_k\|^2}{z_k^T A_k z_k} \\ B_k^T A x_{k+1} &= z_k - t_k A_k z_k \\ r_{k+1} &= A_k z_k \ \text{ normalized} \\ B_{k+1} &= B_k(I - \gamma r_{k+1} r_{k+1}^T). \end{aligned}$$

By definition,

$$\begin{aligned} z_{k+1} &= B_{k+1}^T A x_{k+1} = (I - \gamma r_{k+1} r_{k+1}^T) B_k^T A x_{k+1} \\ &= (I - \gamma r_{k+1} r_{k+1}^T)(z_k - t_k A_k z_k) \\ &= (I - \gamma r_{k+1} r_{k+1}^T)\Big(z_k - \frac{\|z_k\|^2}{r_{k+1}^T z_k} r_{k+1}\Big) \\ &= z_k - \Big(\gamma r_{k+1}^T z_k + (1-\gamma)\frac{\|z_k\|^2}{r_{k+1}^T z_k}\Big) r_{k+1}, \end{aligned}$$

where once again $r_{k+1}^T z_k \neq 0$ because $x_k \neq 0$. Hence we can rewrite the iteration as follows:

$$\begin{aligned} r_{k+1} &= A_k z_k \ \text{ normalized} \\ B_{k+1} &= B_k(I - \gamma r_{k+1} r_{k+1}^T) \\ z_{k+1} &= z_k - \Big(\gamma r_{k+1}^T z_k + (1-\gamma)\frac{\|z_k\|^2}{r_{k+1}^T z_k}\Big) r_{k+1}. \end{aligned}$$

Normalizing each vector z_k to obtain the corresponding unit vector u_k results in the following iteration.

Shor matrix iteration *Given any n-by-n symmetric positive-definite matrix A_0 and any unit vector $u_0 \in \mathbf{R}^n$, compute the following sequences for $k = 1, 2, 3, \ldots$:*

$$\left\{ \begin{array}{rcl} r_k &=& A_{k-1} u_{k-1} \ \text{ normalized} \\ c_k &=& r_k^T u_{k-1} \\ D_k &=& I - \gamma r_k r_k^T \\ A_k &=& D_k A_{k-1} D_k \\ u_k &=& u_{k-1} - \Big(\gamma c_k + \dfrac{1-\gamma}{c_k}\Big) r_k \ \text{ normalized.} \end{array} \right.$$

Note that in the case $\gamma = 0$ (steepest descent), $A_k = A_0$ for all k, while for $\gamma = 1$ (conjugate gradient), $A_n = 0$ and $x_n = 0$.

As noted earlier, we are interested in how the condition number $\kappa(A_k)$ evolves. We begin with the following elementary result for the trace, determinant and condition number.

Proposition 2.1 *We have*

$$\operatorname{tr} A_k < \operatorname{tr} A_{k-1}, \quad \det A_k = (1-\gamma)^2 \det A_{k-1},$$

and

$$\kappa(A_k) \leq \frac{\kappa(A_{k-1})}{(1-\gamma)^2}.$$

Proof The first inequality is immediate, as is the equality, observing that the matrix D_k has one eigenvalue equal to $1-\gamma$ and the rest all one. For the second inequality, letting $\lambda_{\max}(\cdot)$ and $\lambda_{\min}(\cdot)$ denote maximum and minimum eigenvalue respectively, observe that for any vector v,

$$v^T A_k v \geq \lambda_{\min}(A_{k-1}) v^T D_k^2 v \geq \lambda_{\min}(A_{k-1})(1-\gamma)^2 \|v\|^2$$

and

$$v^T A_k v \leq \lambda_{\max}(A_{k-1}) v^T D_k^2 v \leq \lambda_{\max}(A_{k-1}) \|v\|^2.$$

□

From this, we can conclude only that $\kappa(A_k)$ does not grow superlinearly. However, numerical experiments suggest the following conjecture.

Conjecture 2.2 *Given any $\gamma < 1$, initial matrix A_0 and initial vector u_0, the condition numbers of the matrices $A_0, A_1, A_2, \ldots$ generated by the Shor matrix iteration stay bounded.*

Suppose this holds. If we set

$$\bar{\kappa} = \limsup_k \kappa(A_k),$$

our observations above imply that the function values $\frac{1}{2} x_k^T A x_k$ generated by Shor's r-algorithm converge to zero linearly with asymptotic rate

$$\left(\frac{\bar{\kappa} - 1}{\bar{\kappa} + 1}\right)^2.$$

Indeed, experiments suggest a much stronger conjecture.

Conjecture 2.3 *For each dimension $n = 1, 2, 3, \ldots$ and each $\gamma \in (0, 1)$, there exists a finite constant $\rho(n, \gamma)$ associated with the Shor matrix iteration, independent of the initial matrix A_0 and the initial vector u_0, such that the condition number of the iterates A_k satisfy*

$$\limsup_k \kappa(A_k) \leq \rho(n, \gamma).$$

This conjecture would imply that Shor's algorithm converges linearly on convex quadratics at an asymptotic linear rate independent of the initial conditioning of the quadratic. Such a result would interpolate nicely between known results for steepest descent ($\gamma = 0$, for which the conjecture is not true) and conjugate gradient ($\gamma = 1$, which terminates with $A_n = 0$ for all A_0). See Figure 1 for a typical example. The graph plots the condition number of the matrix A_k against the iteration count k, for various choices of γ. The initial matrix A_0 was a Hilbert matrix of size 10, and the initial unit vector u_0 was generated randomly. For values of γ that are very close to 1, the method nearly solves the optimization problem in n steps, but then the next step results in a huge increase in the condition number which is reduced in subsequent iterations. This pattern repeats, suggesting that $\rho(n, \gamma) \to \infty$ as $\gamma \to 1$. On the other hand, while it seems quite possible that $\rho(n, \gamma)$ can be taken arbitrarily close to 1 as $\gamma \to 0$, choosing γ close to 0 is not desirable as the transient decrease in the condition number is slower the closer γ is to 0. These observations motivate a choice of γ that is strictly between 0 and 1.

The trace and determinant of a positive-definite matrix give crude bounds on its condition number, as shown in the following result.

Proposition 2.4 *For any n-by-n symmetric positive definite matrix A, define*

$$\mu(A) \;=\; \frac{1}{n} \frac{\operatorname{tr} A}{(\det A)^{1/n}}.$$

Then

$$1 \;\leq\; \mu(A) \;\leq\; \kappa(A) \;\leq\; 4(\mu(A))^n.$$

Proof The first inequality is just the arithmetic-geometric mean inequality, while the second is immediate. The third inequality follows easily from [MUV+97, Thm 2]. □

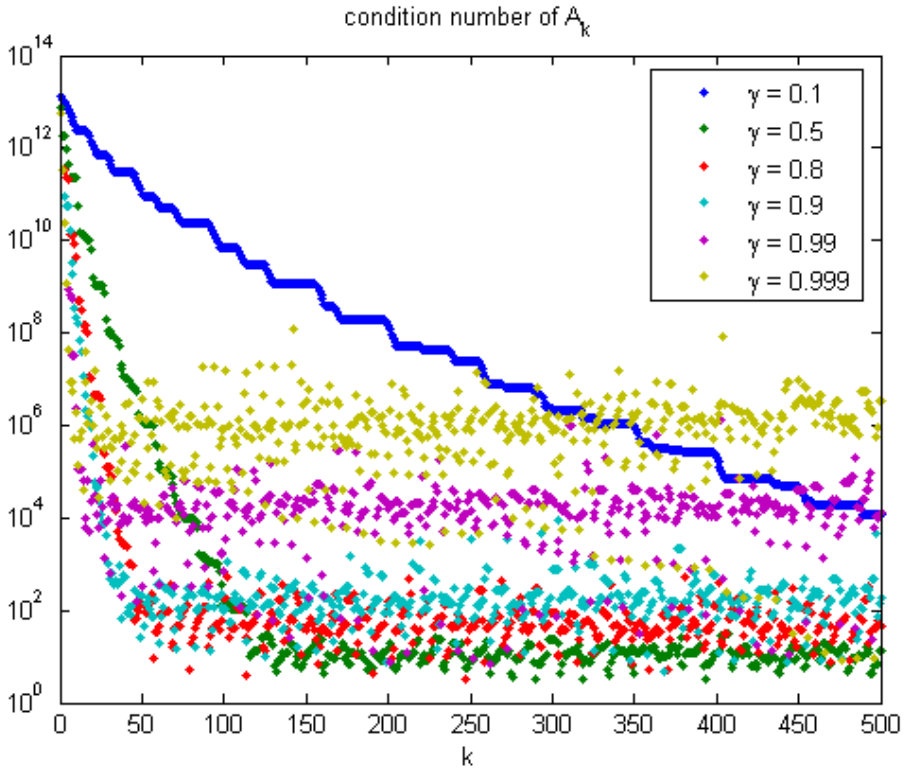

Figure 1: Conditioning of the A_k matrices generated by the Shor algorithm when the initial matrix is the 10 by 10 Hilbert matrix.

In order to keep the presentation self-contained, we also note the following simple proof of a weaker version of the third inequality, replacing the factor 4 by $n^n/(n-1)^{n-1}$. Denote the eigenvalues of A by $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n > 0$. Then

$$\kappa(A) = \frac{\lambda_1}{\lambda_n} \quad \text{and} \quad \mu(A) = \frac{\lambda_1 + \lambda_2 + \cdots + \lambda_n}{n(\lambda_1 \cdot \lambda_2 \cdot \cdots \cdot \lambda_n)^{1/n}}.$$

We have

$$\sum_{i=1}^{n} \lambda_i = n\mu(A) \left(\prod_{i=1}^{n} \lambda_i \right)^{1/n},$$

so dividing by λ_n shows

$$\begin{aligned} \alpha &= \sum_{i=1}^{n-1} \frac{\lambda_i}{\lambda_n} < \sum_{i=1}^{n} \frac{\lambda_i}{\lambda_n} = n\mu(A) \left(\prod_{i=1}^{n} \frac{\lambda_i}{\lambda_n} \right)^{1/n} \\ &= n\mu(A) \left(\prod_{i=1}^{n-1} \frac{\lambda_i}{\lambda_n} \right)^{\frac{1}{n-1}\frac{n-1}{n}} \leq n\mu(A) \left(\frac{\alpha}{n-1} \right)^{\frac{n-1}{n}}, \end{aligned}$$

where the last inequality follows from the arithmetic-geometric mean inequality again. This implies the weaker upper bound on $\kappa(A)$, which is all we need for what follows.

Thus, for fixed n, the condition number $\kappa(A_k)$ remains bounded if and only if $\mu(A_k)$ remains bounded. Since we know that $\det A_k$ decreases by the constant factor $(1-\gamma)^2$ at every step, we can state our conjectures about the condition number in terms of the trace of A_k rather than its condition number. Then Conjecture 2.2 becomes:

Conjecture 2.5 *Given any initial n-by-n matrix A_0 and initial vector u_0, the matrices $A_0, A_1, A_2, \ldots$ generated by the Shor matrix iteration have the property that the quantity*

$$(1-\gamma)^{-\frac{2k}{n}} \operatorname{tr} A_k$$

stays bounded.

The experimental observations of Figure 1 suggest that, as long as γ is not too close to 0 or 1, the conditioning of the matrices A_k generated by the Shor iteration is in some sense "self-correcting": over a long sequence of iterations, any ill-conditioning evolves away, settling into a stable state of

relatively small fluctuations. However, the closer γ is to 1, the less stable is this behaviour. As we observed above, since $\det A_k$ decreases linearly, the behaviour of $\operatorname{tr} A_k$ gives a reasonable measure of the conditioning, and this behaviour suggests a partial explanation for the self-correcting mechanism. Since

$$\operatorname{tr} A_k \;=\; \operatorname{tr}\Big((I-\gamma r_k r_k^T)A_{k-1}(I-\gamma r_k r_k^T)\Big) \;=\; \operatorname{tr} A_{k-1} - \gamma(2-\gamma) r_k^T A_{k-1} r_k,$$

the reduction in trace is least when r_k is close to an eigenvector corresponding to the smallest eigenvalue of A_{k-1}. In this case, since r_k is $A_{k-1}u_{k-1}$ normalized, the unit vectors r_k and u_{k-1} must be close, so the scalar c_k must be close to one, and then the formula for the new unit vector u_k implies that it must be almost orthogonal to u_{k-1}. In particular, the iteration does not allow the vectors r_k to "line up" in the direction of a single eigenvector.

In the two-dimensional case, this self-correcting behaviour is enough to verify Conjecture 2.2. We present a proof in the case $\gamma = 1/2$, depending on the fact that, while the condition number of A_k can increase from one iteration to the next, after *two* iterations it must decrease.

Theorem 2.6 *For the Shor matrix iteration in dimension $n = 2$ and with constant $\gamma = 1/2$, for any step k such that the matrices A_{k-1} and A_{k+1} are both defined,*

$$\kappa(A_{k+1}) < \kappa(A_{k-1}).$$

Consequently, the Shor r-algorithm (with $\gamma = 1/2$) for minimizing a two-variable strictly convex quadratic either terminates or converges linearly.

Proof Since the function $t \mapsto t^{\frac{1}{2}} + t^{-\frac{1}{2}}$ is increasing for $t \geq 1$, it suffices to prove

$$\sqrt{\kappa(A_{k+1})} + \frac{1}{\sqrt{\kappa(A_{k+1})}} \;<\; \sqrt{\kappa(A_{k-1})} + \frac{1}{\sqrt{\kappa(A_{k-1})}},$$

or equivalently

$$\frac{\operatorname{tr} A_{k+1}}{\sqrt{\det A_{k+1}}} \;<\; \frac{\operatorname{tr} A_{k-1}}{\sqrt{\det A_{k-1}}}.$$

By our observations above,

$$\det A_{k+1} = \frac{1}{16} \det A_{k-1}.$$

On the other hand,

$$\begin{aligned} \operatorname{tr} A_k &= \operatorname{tr} A_{k-1} - \frac{3}{4} r_k^T A_{k-1} r_k \\ \operatorname{tr} A_{k+1} &= \operatorname{tr} A_k - \frac{3}{4} r_{k+1}^T A_k r_{k+1}. \end{aligned}$$

Hence we want to show

$$\operatorname{tr} A_{k-1} \;>\; 4\operatorname{tr} A_{k+1} \;=\; 4\Big(\operatorname{tr} A_{k-1} - \frac{3}{4}(r_k^T A_{k-1} r_k + r_{k+1}^T A_k r_{k+1})\Big),$$

or in other words

$$r_k^T A_{k-1} r_k + r_{k+1}^T A_k r_{k+1} \;>\; \operatorname{tr} A_{k-1}.$$

Let us summarize our task, in simplified notation. Given any unit vector $u \in \mathbf{R}^2$ (formerly u_{k-1}) and any 2-by-2 symmetric positive-definite matrix F (formerly A_{k-1}), we define

$$\begin{aligned} r &= \frac{1}{\|Fu\|} Fu \\ c &= r^T u \\ D &= I - \frac{1}{2} r r^T \\ G &= DFD \\ v &= u - \frac{1}{2}\Big(c + \frac{1}{c}\Big) r \\ w &= \frac{1}{\|Gv\|} Gv. \end{aligned}$$

In our former notation, $r = r_k$, $c = c_k$, $D = D_k$, $G = A_k$, v is some positive multiple of the unit vector u_k, and $w = r_{k+1}$. We want to show the inequality

$$r^T F r + w^T G w > \operatorname{tr} F,$$

or in other words

$$r^T F r + \frac{v^T G^3 v}{v^T G^2 v} > \operatorname{tr} F.$$

Without loss of generality we can assume $r = [1, 0]^T$. Indeed, if we prove the result in this special case, we can deduce the case for general r by a simple change of variables: choose any orthogonal matrix U satisfying $Ur = [1, 0]^T$,

and then apply the special case with the vector u replaced by Uu and the matrix F replaced by UFU^T.

So, we can assume $r = [1, 0]^T$ and then, by the definition of c, we must have $u = [c, s]^T$ where $c^2 + s^2 = 1$. Notice $c > 0$ since F is positive definite. Without loss of generality, by rescaling if necessary, we can assume the bottom row of the matrix F has norm one, and then we must have

$$F = \begin{bmatrix} a & -s \\ -s & c \end{bmatrix} \quad \text{for some } a > 0.$$

We deduce

$$G \;=\; \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a & -s \\ -s & c \end{bmatrix} \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & 1 \end{bmatrix} \;=\; \begin{bmatrix} \frac{a}{4} & -\frac{s}{2} \\ -\frac{s}{2} & c \end{bmatrix}.$$

We want to show

$$a + \frac{v^T G^3 v}{v^T G^2 v} > a + c,$$

or in other words

$$v^T G^3 v > c v^T G^2 v.$$

Since

$$v = \begin{bmatrix} c \\ s \end{bmatrix} - \frac{1}{2}\Big(c + \frac{1}{c}\Big) \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

we deduce $2cv = s[-s, 2c]^T$, so

$$8cGv = (4G)(2cv) = s \begin{bmatrix} -s(a + 4c) \\ 2 + 6c^2 \end{bmatrix}.$$

Finally, we have

$$\begin{aligned} 256c^2(v^T G^3 v - c v^T G^2 v) \;&=\; (8cGv)^T (4G)(8cGv) - 4c(8cGv)^T (8cGv) \\ &= (8cGv)^T (4(G - cI))(8cGv) \\ &= s^2 \begin{bmatrix} -s(a + 4c) \\ 2 + 6c^2 \end{bmatrix}^T \begin{bmatrix} a - 4c & -2s \\ -2s & 0 \end{bmatrix} \begin{bmatrix} -s(a + 4c) \\ 2 + 6c^2 \end{bmatrix} \\ &= s^2\Big(s^2(a - 4c)(a + 4c)^2 + 4s^2(a + 4c)(2 + 6c^2)\Big) \\ &= s^4(a + 4c)(a^2 + 8c^2 + 8) \;>\; 0 \end{aligned}$$

as required, using the fact that $s \neq 0$ since $v \neq 0$ by assumption. □

Numerical experiments show that it is not always the case that $\kappa(A_{k+n-1}) < \kappa(A_{k-1})$ for $n > 2$, so proving linear convergence for $n > 2$ will require a different approach. Nonetheless, both the observations of Figure 1 and the relationship with the conjugate gradient method indicate that whatever result might be established, it will probably involve a characterization of behaviour over n steps.

3 Concluding Remarks

Our interest in Shor's r-algorithm has two different motivations. One is its apparently substantial practical success in minimizing nonsmooth functions. The other is that the algorithm interpolates between two pillars of optimization, steepest descent and conjugate gradient, and seems to have interesting convergence properties that remain to be established. We hope that our analysis of the r-algorithm in the simplest setting imaginable will stimulate further research on its theoretical behaviour – perhaps by the master of nonlinear optimization convergence analysis, M.J.D. Powell? Happy Birthday, Mike!

References

[KK00] F. Kappel and A. Kuntsevich. An implementation of Shor's r-algorithm. *Computational Optimization and Applications*, 15:193–205, 2000.

[Lue84] D.G. Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley, Reading, MA, 1984.

[MUV+97] J.K. Merikoski, U. Urpala, A. Virtanen, T.-Y. Tam, and F. Uhlig. A best upper bound for the 2-norm condition number of a matrix. *Linear Algebra and its Applications*, 254:355–365, 1997.

[Sho85] N.Z. Shor. *Minimization Methods for Non-differentiable Functions*. Springer-Verlag, New York, 1985.

[Tod86] M.J. Todd. The symmetric rank-one quasi-Newton algorithm is a space-dilation subgradient algorithm. *Operations Research Letters*, 5:217–219, 1986.