

Minimizing convex quadratics with variable precision Krylov methods

Philippe Toint

(with Serge Gratton, Ehouarn Simon and D. Titley-Peloquin)



Namur Center for Complex Systems (naXys), University of Namur, Belgium
CIMI, INP, Toulouse, France
McGill University, Montreal, Canada

(`philippe.toint@unamur.be`)

Numerical Analysis Seminar, Oxford, November 2019

Thanks

- CIMI, Institut National Polytechnique, Toulouse, France (ANR-11-IDEX-0002-02)
- University of Namur, Belgium
- Belgian National Fund for Scientific Research

The (simple?) problem

We consider the unconstrained quadratic optimization (QO) problem:

$$\text{minimize } q(x) = \frac{1}{2}x^T Ax - b^T x$$

for $x, b \in \mathbb{R}^n$ and A an $n \times n$ symmetric positive-definite matrix.

A truly “core” problem in optimization (and linear algebra)

- the simplest nonlinear optimization problem
- subproblem in many methods for general nonlinear unconstrained optimization
- central in linear algebra (including solving elliptic PDEs)

Working assumptions

For what follows, we assume that

- the problem size n is large enough and A is dense enough to make **factorization of A unavailable**
- a Krylov iterative method (**Conjugate Gradients**) is used
- the **cost** of running this iterative method is **dominated by the products Av**

Focus on an **optimization point of view**: look at decrease in q rather than at decrease in the associated system's residual

ex: ensuring **sufficient decrease** in trust-region methods

Our aim, for x_* solution of QO,

$$\text{Find } x_k \text{ such that } |q(x_k) - q(x_*)| \leq \epsilon |q(x_0) - q(x_*)|.$$

A first motivating example: weather forecasting (1)

The weakly-constrained 4D-Var formulation:

$$\min_{x \in \mathbb{R}^n} \frac{1}{2} \|x_0 - x_b\|_{B^{-1}}^2 + \frac{1}{2} \sum_{j=0}^N \|\mathcal{H}_j(x_j) - y_j\|_{R_j^{-1}}^2 + \frac{1}{2} \sum_{j=1}^N \underbrace{\|x_j - \mathcal{M}_j(x_{j-1})\|_{Q_j^{-1}}^2}_{q_j}$$

- $x = (x_0, \dots, x_N)^T$ is the **state** control variable (with $x_j = x(t_j)$)
- x_b is the background given at the initial time (t_0).
- $y_j \in \mathbb{R}^{m_j}$ is the observation vector over a given time interval
- \mathcal{H}_j maps the state vector x_j from model space to observation space
- \mathcal{M}_j is an integration of the **numerical model** from time t_{j-1} to t_j
- B , R_j and Q_j are the covariance matrices of background, observation and model error. **B and Q_j impractical to "invert"**

A first motivating example: weather forecasting (2)

Solve by a Gauss-Newton method whose subproblem (at iteration k) is

$$\min_{\delta x} \frac{1}{2} \|\delta x_0 - b^{(k)}\|_{\mathbf{B}^{-1}}^2 + \frac{1}{2} \sum_{j=0}^N \left\| H_j^{(k)} \delta x_j - d_j^{(k)} \right\|_{\mathbf{R}_j^{-1}}^2 + \frac{1}{2} \sum_{j=1}^N \underbrace{\|\delta x_j - M_j^{(k)} \delta x_{j-1} - c_j^{(k)}\|_{\mathbf{Q}_j^{-1}}^2}_{\delta q_j}$$

- δx is the increment in x .
- The vectors $b^{(k)}$, $c_j^{(k)}$ and $d_j^{(k)}$ are defined by

$$b^{(k)} = x_b - x_0^{(k)}, \quad c_j^{(k)} = q_j^{(k)}, \quad d_j^{(k)} = \mathcal{H}_j(x_j^{(k)}) - y_j$$

and are calculated at the outer loop.

A first motivating example: weather forecasting (3)

Can be rewritten as

$$\min_{\delta x} q_{\text{st}} = \frac{1}{2} \|L\delta x - b\|_{D^{-1}}^2 + \frac{1}{2} \|H\delta x - d\|_{R^{-1}}^2$$

where

$$\bullet L = \begin{pmatrix} I & & & & \\ -M_1 & I & & & \\ & -M_2 & I & & \\ & & \ddots & \ddots & \\ & & & -M_N & I \end{pmatrix}$$

- $d = (d_0, d_1, \dots, d_N)^T$ and $b = (b, c_1, \dots, c_N)^T$
- $H = \text{diag}(H_0, H_1, \dots, H_N)$
- $D = \text{diag}(B, Q_1, \dots, Q_N)$ and $R = \text{diag}(R_0, R_1, \dots, R_N)$

A first motivating example: weather forecasting (3)

$$\min_{\delta x} q_{st} = \frac{1}{2} \|L\delta x - b\|_{D^{-1}}^2 + \frac{1}{2} \|H\delta x - d\|_{R^{-1}}^2$$

This is a standard QO, but **HUGE!** Note that

$$\nabla^2 q_{st} = L^T D^{-1} L + H^T R^{-1} H$$

In addition $D^{-1} = \text{diag}(B^{-1}, Q_1^{-1}, \dots, Q_N^{-1})$ is unavailable!

Thus $\nabla^2 q_{st} v$ (a Hessian times vector product) must be computed by

- $w = Lv$,
- solve $Dz = w$ using some (preconditioned) Krylov method
- $v = L^T z + H^T R^{-1} H v$

A second motivating example: variable precision arithmetic

Next barrier in **hyper computing**: energy dissipation!

Heat production is proportional to chip surface, hence

$$\text{energy output} \approx (\text{number of digits used})^2$$

Architectural trend: use multiprecision arithmetic

- graphical processing units (GPUs)
- hierarchy of specialized CPUs (double, single, half, ...)

How to use this hierarchy optimally for fully accurate results?

Inaccuracy frameworks

Our proposal;

Make the Krylov methods for QO more efficient by allowing error on the matrix-vector product (the dominant computation)

Two frameworks of interest:

- Continuous accuracy levels

ex: WC-4D-VAR, where accuracy in the inversion $Dz = w$ can be continuously chosen

- Discrete accuracy levels

ex: double-single-half precision arithmetic

Considered here:

- Conjugate Gradients (CG)

with (wlog) $x_0 = 0$ and $q(x_0) = 0$.

A central equality

Define $r(x) \stackrel{\text{def}}{=} Ax - b = \nabla q(x)$ and $Ax_* = b$.

$$q(x) - q(x_*) = \frac{1}{2} \|r(x)\|_{A^{-1}}^2$$

$$\begin{aligned} \frac{1}{2} \|r(x)\|_{A^{-1}}^2 &= \frac{1}{2} (Ax - b)^T A^{-1} (Ax - b) \\ &= \frac{1}{2} (x - x_*)^T A (x - x_*) \\ &= \frac{1}{2} (x^T Ax - 2x^T Ax_* + x_*^T Ax_*) \\ &= q(x) - q(x_*) \end{aligned}$$

Hence

Decrease in q can be monitored by considering **the A^{-1} norm** of its gradient

The primal-dual norm

⇒ natural to consider the inaccuracy on the product Av by measuring the backward error

$$\|E\|_{A^{-1},A} \stackrel{\text{def}}{=} \sup_{x \neq 0} \frac{\|Ex\|_{A^{-1}}}{\|x\|_A} = \|A^{-1/2}EA^{-1/2}\|_2$$

(primal-dual norm)

Let A be a symmetric and positive definite matrix and E be any symmetric perturbation. Then, if $\|E\|_{A^{-1},A} < 1$, the matrix $A + E$ is symmetric positive definite.

The main idea

Krylov methods **reduce the (internally recurred) residual r_k** on successive nested Krylov spaces

⇒ can expect r_k to converge to zero

⇒ keep $r(x_k) - r_k$ small in the appropriate norm

For CG, if

$$\max \left[\|r_k - r(x_k)\|_{A^{-1}}, \|r_k\|_{A^{-1}} \right] \leq \frac{\sqrt{\epsilon}}{2} \|b\|_{A^{-1}}$$

then

$$|q(x_k) - q(x_*)| \leq \epsilon |q(x_*)|$$

The inexact Conjugate Gradients algorithm

Theoretical inexact CG algorithm

1. Set $x_0 = 0$, $\beta_0 = \|b\|_2^2$, $r_0 = -b$ and $p_0 = r_0$
2. For $k=0, 1, \dots$, do
3. $c_k = (A + E_k)p_k$
4. $\alpha_k = \beta_k / p_k^T c_k$
5. $x_{k+1} = x_k + \alpha_k p_k$
6. $r_{k+1} = r_k + \alpha_k c_k$
7. if r_{k+1} is small enough then stop
8. $\beta_{k+1} = r_{k+1}^T r_{k+1}$
9. $p_{k+1} = -r_{k+1} + (\beta_{k+1} / \beta_k) p_k$
10. EndFor

Results for the inexact CG

Let $\epsilon > 0$ and let $\phi \in \mathbb{R}_+^k$ such that $\sum_{j=1}^k \phi_j^{-1} \leq 1$. Suppose also that, for all $j \in \{0, \dots, k-1\}$,

$$\|E_j\|_{A^{-1}, A} \leq \omega_j^{\text{CG}} \stackrel{\text{def}}{=} \frac{\sqrt{\epsilon} \|b\|_{A^{-1}} \|p_j\|_A}{2\phi_{j+1} \|r_j\|_2^2 + \sqrt{\epsilon} \|b\|_{A^{-1}} \|p_j\|_A} \quad (0.1)$$

Then

$$\|r(x_k) - r_k\|_{A^{-1}} \leq \frac{1}{2} \sqrt{\epsilon} \|b\|_{A^{-1}}.$$

and

$$|q(x_k) - q(x_*)| \leq \frac{1}{2} \sqrt{\epsilon} |q(x_*)|$$

Achieved vs optimal decrease

Let q be the value of the quadratic recurred internally by CG.

Let x be the result of applying the CG algorithm with inexact products and suppose that the above error bounds hold. Then

$$\frac{|q(x) - q|}{|q(x_*)|} \leq \frac{1}{2}\sqrt{\epsilon}(1 + \sqrt{\epsilon}).$$

Can one trust the internally computed decrease? Rather **pessimistic!**

Managing the inaccuracy budget

Assume k_{\max} , an estimate of the maximum number of iterations, is known.

At iteration j of CG:

$$\left. \begin{array}{l} v_j \\ r_{j-1} \\ \phi_j \end{array} \right\} \rightarrow \left. \begin{array}{l} v_j \\ \omega_j \end{array} \right\} \rightarrow \text{product routine} \rightarrow \frac{(A + E_j)v_j}{\|E_j\|_{A^{-1},A}} \rightarrow \hat{\phi}_j \rightarrow \phi_{j+1}$$

where

$$\hat{\phi}_j^{\text{CG}} = \frac{(1 - \|E_j\|_{A^{-1},A}) \frac{1}{2} \sqrt{\epsilon} \|b\|_{A^{-1}} \|p_j\|_A}{\|E_j\|_{A^{-1},A} \|r_j\|_2^2} \quad \text{and} \quad \phi_{j+1} = \frac{k_{\max} - j}{1 - \sum_{p=1}^j \hat{\phi}_p^{-1}}$$

So what?

- The primal-dual norm $\|E_j\|_{A^{-1},A}$ is sometimes **difficult to evaluate**
- The **error bounds** remain unfortunately **impractical** (they involve $\|b\|_{A^{-1}}$, $\|v_j\|_A$ or $\|p_j\|_A$, which cannot be computed readily in the course of the CG algorithm).
- The **termination test** $\|r_k\|_{A^{-1}} \leq \frac{1}{2}\sqrt{\epsilon} \|b\|_{A^{-1}}$ also involves the unavailable $\|r_k\|_{A^{-1}}$

Give up? Not quite...

- the CG error bound allows a **growth** of the order of $\|r_j\|^{-2} \|p_j\|_A$
- The ϕ_j may be viewed as an **error management strategy**. A simple choice is to define $\phi_j = n$ for all j but there may be better options (discussed later).

Adhoc approximations (1)

Abandon theoretical but unavailable quantities \rightarrow **approximate** them:

- $\|E\|_{A^{-1},A} \leq \lambda_{\min}(A)^{-1} \|E\|_2$
- $\|p\|_A \approx \sqrt{\frac{1}{n} \text{Tr}(A)} \|p\|_2$
(ok for p with random independent components)
- $\|b\|_{A^{-1}} = \sqrt{2|q(x_*)|} \approx \sqrt{2|q_k|} \approx \sqrt{|b^T x_k|}$

Replace

$$\|E_j\|_{A^{-1},A} \leq \frac{\sqrt{\epsilon} \|b\|_{A^{-1}} \|p_j\|_A}{2\phi_{j+1} \|r_j\|_2^2 + \sqrt{\epsilon} \|b\|_{A^{-1}} \|p_j\|_A}$$

by

$$\frac{\|E_j\|_2}{\lambda_{\min}(A)} \leq \frac{\sqrt{\epsilon} \sqrt{|q_j|} \sqrt{\text{Tr}(A)} \|p_j\|_2}{\sqrt{2n} \phi_{j+1} \|r_j\|_2^2 + \sqrt{\epsilon} \sqrt{|q_j|} \sqrt{\text{Tr}(A)} \|p_j\|_2},$$

Adhoc approximations (2)

Termination test (Arioli & Gratton):

$$\|r_k^{-1}\|_{A^{-1}}^2 \approx q_{k-d} - q_k \leq \frac{1}{4}\epsilon |q_k|$$

for some **stabilization delay** d (e.g. 10)

Using the true (unavailable) quantities (1)

Would this work at all if using the **true** $\|b\|_{A^{-1}}$, $\|v_j\|_A$ and $\|p_j\|_A$?

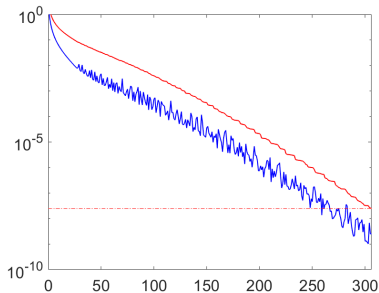
Start with continuous accuracy levels: note that

Continuous accuracy levels \Rightarrow no room for inaccuracy budget management!

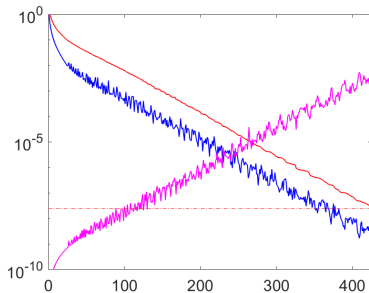
Continuous accuracy levels (1)

$\text{diag}(\text{logspace}(-4, 0, 100))$ ($n = 100$, $\kappa = 10^4$)

(a) double precision CG



(b) theoretical inexact CG

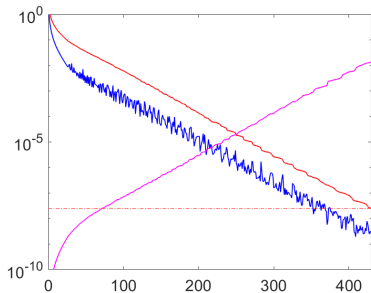


— Euclidean norm of residual
— A^{-1} norm of residual
— ω_j
- - - tolerance

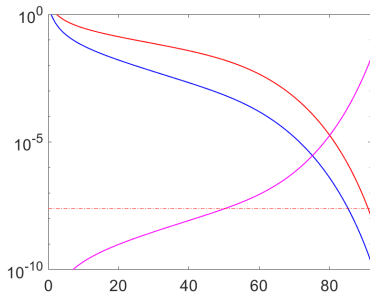
Continuous accuracy levels (2)

$\text{diag}(\text{logspace}(-4, 0, 100))$ ($n = 100$, $\kappa = 10^4$)

(c) practical inexact CG



(d) with reorthogonalization

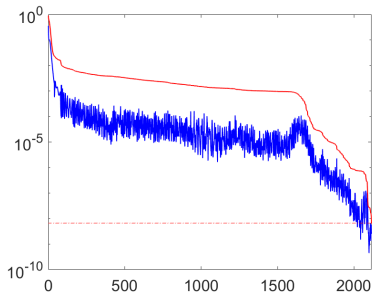


— Euclidean norm of residual
— A^{-1} norm of residual
— ω_j
- - - tolerance

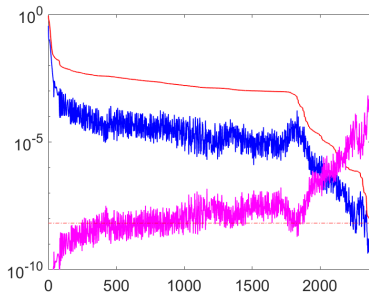
Continuous accuracy levels (3)

nos1.mat ($n = 237$, Matrix market, scaled to unit norm, $\kappa = 10^8$)

(a) double precision CG



(b) theoretical inexact CG

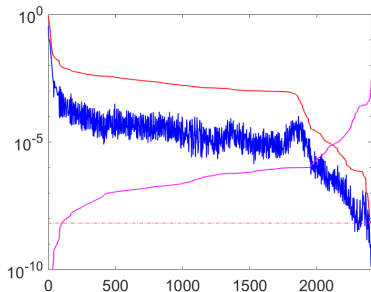


- Euclidean norm of residual
- A^{-1} norm of residual
- ω_j
- - - tolerance

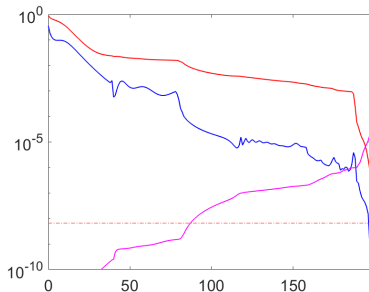
Continuous accuracy levels (4)

nos1.mat ($n = 237$, Matrix market, scaled to unit norm, $\kappa = 10^8$)

(c) practical inexact CG



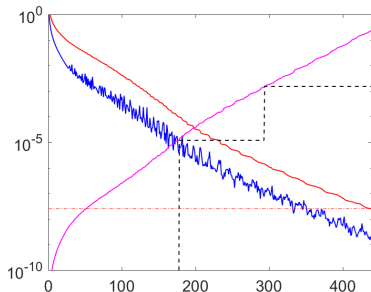
(d) with reorthogonalization



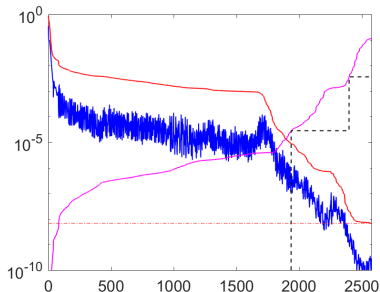
- Euclidean norm of residual
- A^{-1} norm of residual
- ω_j
- - - tolerance

Practical inexact CG with discrete accuracy levels

(a) `diag(logspace(-4, 0, 100))`



(b) `nos1.mat`



- Euclidean norm of residual
- A^{-1} norm of residual
- ω_j
- - $\hat{\omega}_j$
- - - tolerance

More numerical tests (1)

- Compare **equivalent numbers of full accuracy products**:
 - Assume **obtaining a full accuracy product is a linearly convergent process of rate ρ**
(realistic for our **weather prediction data assimilation example**)
 - Cost of an fully accurate matrix-vector product:

$$\frac{\log(\epsilon_M)}{\log(\rho)}$$

- Cost of an ω -accurate matrix-vector product:

$$\frac{\log(\omega)}{\log(\rho)}$$

⇒ sum these values during computing (grey bar)

- Also compare numbers of iterations for convergence to machine precision (black bars).

More numerical tests (2)

Consider 4 algorithms:

CG: the standard full-accuracy CG

iCG: the inexact CG (with exact bounds, for now)

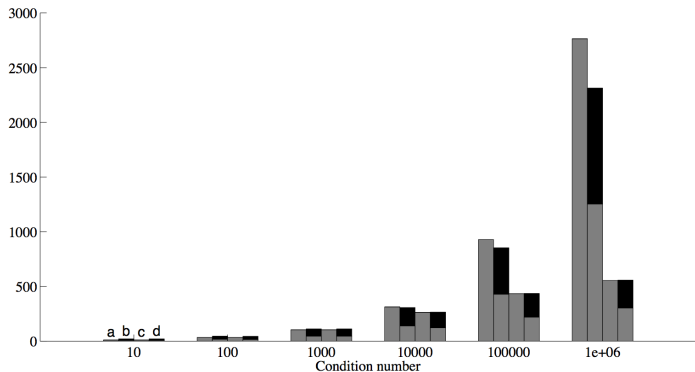
CGR: the full-accuracy CG *with reorthogonalization*

iCGR: the inexact CGR (with exact bounds, for now)

(from left to right: CG, iCG, CGR, iCGR and increasing condition number)

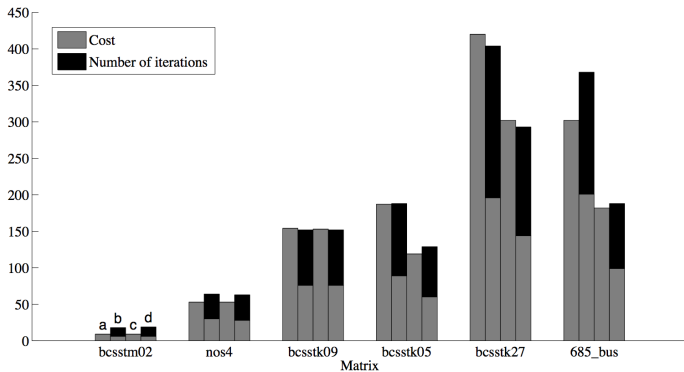
Continuous accuracy levels (1)

Synthetic matrices



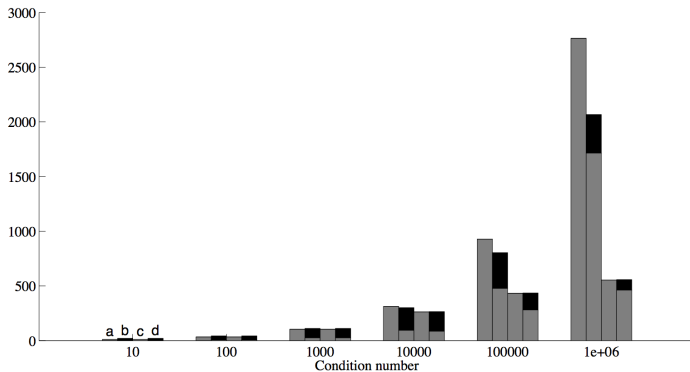
Continuous accuracy levels (2)

Matrix market matrices



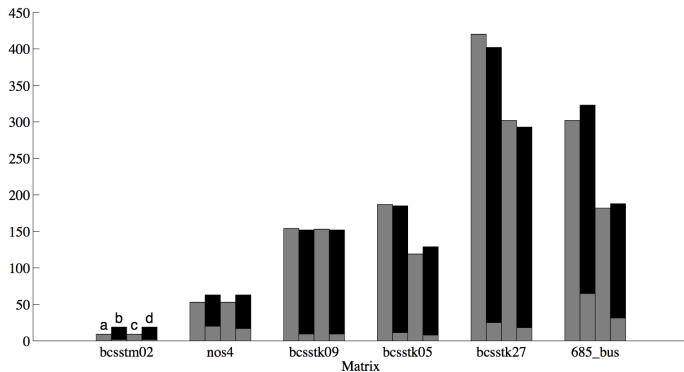
Discrete accuracy levels (1)

Synthetic matrices



Discrete accuracy levels (2)

Matrix market matrices



Conclusions and perspectives

Summary:

- Optimization-focused theory for iterative QO with inexact products
- Theoretical gains substantial
- Translates well to practice after approximations

Perspectives:

- More general (controlable) inexactness in optimization (inexactly weighted least-squares, ...)

Thank your for your attention!