Optimization with inexact gradient and function

Serge Gratton with E. Simon, D. Titley-Peloquin and P. Toint

University of Toulouse and IRIT, France

ANITI
(serge.gratton@toulouse-inp.fr)

SFO, November 2020

Thanks

- 3IA Artificial and Natural Intelligence Toulouse Institute
- CIMI, Institut National Polytechnique, Toulouse, France (ANR-11-IDEX-0002-02)

Outline for section 1

Why multiprecision?

Paraphrasing [Higham, 2017]:

- Variable precision is becoming more and more accessible in hardware and software.
- Using lower precision can drastically reduce computational running time (e.g. IEEE single up to 14 times faster than IEEE double).
- Our challenge is to better understand the accuracy of algorithms in low precision.

How does multiprecision arithmetic affect the convergence rate and final accuracy of minimization algorithms?

Why multiprecision?

Paraphrasing [Higham, 2017]:

- Variable precision is becoming more and more accessible in hardware and software.
- Using lower precision can drastically reduce computational running time (e.g. IEEE single up to 14 times faster than IEEE double).
- Our challenge is to better understand the accuracy of algorithms in low precision.

How does multiprecision arithmetic affect the convergence rate and final accuracy of minimization algorithms?

The (simple?) problem

We consider the unconstrained quadratic optimization (QO) problem:

$$\text{minimize} \quad q(x) = \tfrac{1}{2} x^T A x - b^T x$$

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

$\boxed{\text{A truly ``core'' problem in optimization (and linear algebra)}}$

- 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, FOM) is used
- the cost of running this iterative method is dominated by the products Av

Focus on an $\boxed{\text{optimization point of view}}$: look at decrease in q rather than at decrease in the associated system's residual

ex: ensuring increase in the likelihood in statistics

Our aim, for x_* solution of QO,

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

A first motivating example: weather forecasting (1)

The $\boxed{\text{weakly-constrained 4D-Var}}$ formulation (See [Y. Tremolet 2006, 2007,..])

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

- $x = (x_0, \ldots, x_N)^T$ 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

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

- 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

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

where

- $L = \begin{pmatrix} I & & & & \\ -M_1 & I & & & \\ & -M_2 & I & & \\ & & \ddots & \ddots & \\ & & & -M_N & I \end{pmatrix}$
- $d = (d_0, d_1, \ldots, d_N)^T$ and $b = (b, c_1, \ldots, c_N)^T$
- $H = \mathrm{diag}(H_0, H_1, \ldots, H_N)$
- $D = \mathrm{diag}(B, Q_1, \ldots, Q_N)$ and $R = \mathrm{diag}(R_0, R_1, \ldots, R_N)$

A first motivating example: weather forecasting (3)

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

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

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

In addition $D^{-1} = \mathrm{diag}(B^{-1}, Q_1^{-1}, \ldots, Q_N^{-1})$ 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

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

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?

Outline for section 2

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:

> - Full orthonormalisation method (FOM)
> - 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 $Ax_* = b$.

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

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

Hence

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

The primal-dual norm

$\Rightarrow$ natural to consider the inaccuracy on the product Av by measuring the backward error

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

(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
$\Rightarrow$ can expect r_k to converge to zero
$\Rightarrow$ keep $r(x_k) - r_k$ small in the appropriate norm

From $q(x) - q(x_*) = \frac{1}{2}\|r(x)\|^2_{A^{-1}}$, $q(x_*) = -\frac{1}{2}\|b\|^2_{A^{-1}}$, and triangular inequality,

For any r_k, 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, . . . , do
3. $\quad c_k = (A + E_k)p_k$
4. $\quad \alpha_k = \beta_k / p_k^T c_k$
5. $\quad x_{k+1} = x_k + \alpha_k p_k$
6. $\quad r_{k+1} = r_k + \alpha_k c_k$
7. $\quad$ if r_{k+1} is small enough then stop
8. $\quad \beta_{k+1} = r_{k+1}^T r_{k+1}$
9. $\quad p_{k+1} = -r_{k+1} + (\beta_{k+1}/\beta_k)p_k$
10. EndFor

Results for the inexact CG

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

$$\|E_j\|_{A^{-1},A} \leq \omega_j^{\mathrm{CG}} \stackrel{\mathrm{def}}{=} \frac{\epsilon_\pi \, \|b\|_{A^{-1}} \|p_j\|_A}{\phi_{j+1} \|r_j\|_2^2 + \epsilon_\pi \, \|b\|_{A^{-1}} \|p_j\|_A} \tag{2.1}$$

Then

$$\|r(x_k) - r_k\|_{A^{-1}} \leq \epsilon_\pi \, \|b\|_{A^{-1}}.$$

Let $\epsilon > 0$ and suppose that, at iteration $k > 0$ of the CG algorithm,

$$\|r_k\|_{A^{-1}} \leq \tfrac{1}{2}\sqrt{\epsilon} \, \|b\|_{A^{-1}}$$

and the product error matrices E_j satisfy (2.1) with $\epsilon_\pi = \frac{1}{2}\sqrt{\epsilon}$ for some $\phi \in \mathbb{R}^k$ (as above). Then

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

The inexact FOM algorithm

Theoretical inexact FOM algorithm

1. Set $\beta = \|b\|_2$, and $v_1 = [b/\beta]$,
2. For k=1, 2, . . . , do
3. $\quad w_k = (A + E_k)v_k$
4. $\quad$ For $i = 1, \ldots, k$ do
5. $\quad\quad h_{i,k} = v_i^T w_k$
6. $\quad\quad w_k = w_k - h_{i,k} v_i$
7. $\quad$ EndFor
8. $\quad h_{k+1,k} = \|w_k\|_2$
9. $\quad y_k = H_k^{-1}(\beta e_1)$
10. $\quad$ if $|h_{k+1,k} e_k^T y_k|$ is small enough then go to 13
11. $\quad v_{k+1} = w_k / h_{k+1,k}$
12. EndFor
13. $x_k = V_k y_k$

Results for the inexact FOM

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

$$\|E_j\|_{A^{-1},A} \leq \omega_j^{\mathrm{FOM}} \stackrel{\mathrm{def}}{=} \min\left[1, \frac{\epsilon_\pi \, \|b\|_{A^{-1}}}{\phi_j \|v_j\|_A \|H_k^{-1}\|_2 \|r_{j-1}\|_2}\right] \quad (2.2)$$

Then

$$\|r(x_k) - r_k\|_{A^{-1}} \leq \epsilon_\pi \, \|b\|_{A^{-1}}.$$

Let $\epsilon > 0$ and suppose that, at iteration $k > 0$ of the FOM algorithm,

$$\|r_k\|_{A^{-1}} \leq \tfrac{1}{2}\sqrt{\epsilon}\,\|b\|_{A^{-1}}$$

and the product error matrices E_j satisfy (2.2) with $\epsilon_\pi = \frac{1}{2}\sqrt{\epsilon}$ for some $\phi \in \mathbb{R}^k$ (as above). Then

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

Using the true quantities (1)

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

Consider 6 algorithms:

- FOM: the standard full-accuracy FOM
- iFOM: the inexact FOM (with exact bounds, for now)
- CG: the standard full-accuracy CG
- CGR: the full-accuracy $\boxed{\text{CG } \textit{with reorthogonalization}}$
- iCG: the inexact CG (with exact bounds, for now)
- iCGR: the inexact CGR (with exact bounds, for now)

Continuous accuracy levels (1)

Comparing equivalent numbers of full accuracy products:

- Assume obtaining full accuracy is a linearly convergent process of rate ρ
  (realistic for our weather prediction data assimilation example)
- Cost of an ϵ-accurate solution:

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

$\Rightarrow$ sum these values during computing and compare them.

Continuous accuracy levels (2)

Compare on:

- synthetic matrices of size 1000×1000 with varying conditioning (from 10^1 to 10^8) and log-linearly spaced eigenvalues
- "real" matrices from the NIST Matrix Market
- use different levels of final accuracy ($\epsilon = 10^{-3},\ 10^{-5}$)

Note that

Continuous accuracy levels

Continuous accuracy levels (3)

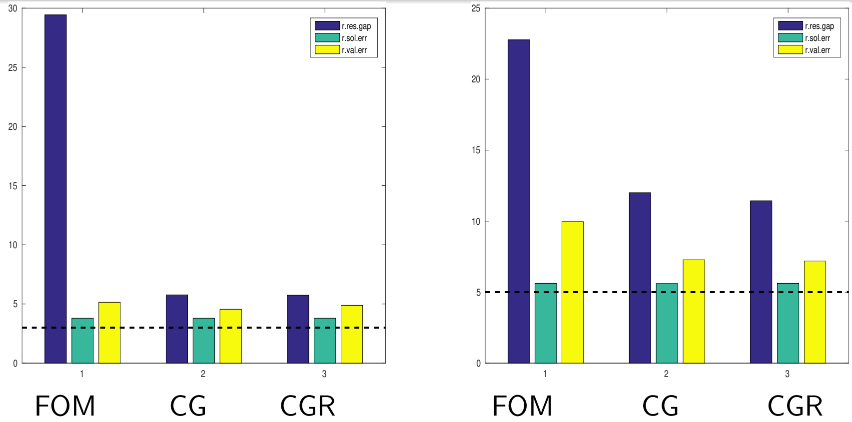


Figure: Exact bounds, $\kappa(A) = 10^1$, $\epsilon = 10^{-3}$ (left), $\kappa(A) = 10^5$, $\epsilon = 10^{-5}$ (right); continuous case

Want blue (gap) and green (stopping criterion error on the quadratic) not worse than epsilon, and yellow (approximate error on the quadratic) close

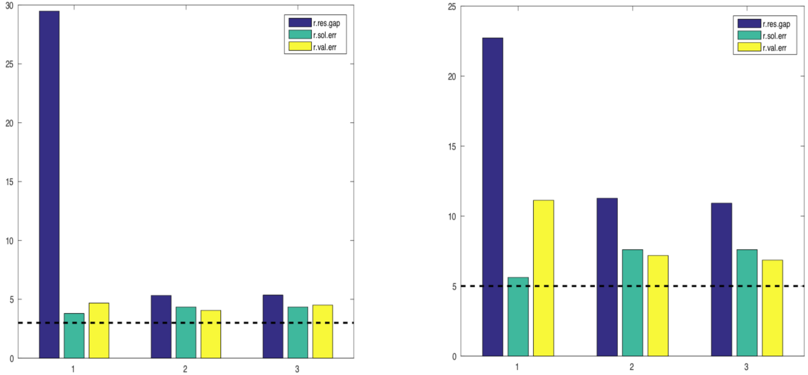
Multiprecision (1)

Focus on $\boxed{\text{multiprecision arithmetic}}$. Assume

- 3 levels of accuracy (double, single, half)
- a ratio of 4 in efficiency when moving from one level to the next

Use the sames matrices and final accuracies as above.

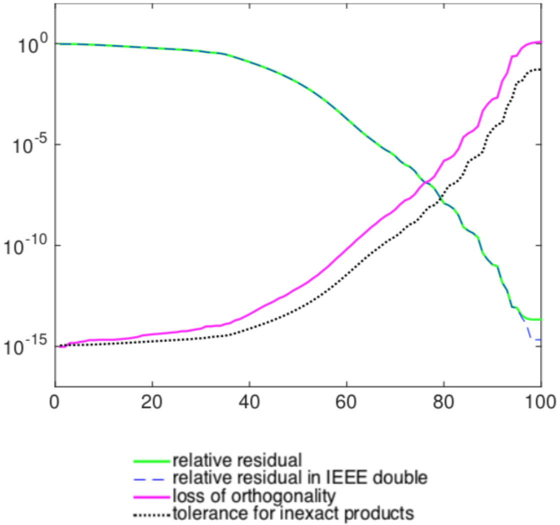
Multiprecision (2)



FOM CG CGR FOM CG CGR

Figure: Exact bounds, $\kappa(A) = 10^1$, $\epsilon = 10^{-3}$ (left), $\kappa(A) = 10^5$, $\epsilon = 10^{-5}$ (right); discontinuous case

An beyond : inexact scalar products



Just relax !

Perfect in theory but...

- The primal-dual norm $\|E_j\|_{A^{-1},A}$ is sometimes difficult to evaluate
- The error bounds remain unfortunately hard to estimate (they involve $\|b\|_{A^{-1}}$, $\|v_j\|_A$ or $\|p_j\|_A$, which cannot be computed readily in the course of the FOM or 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 FOM error bound allows a growth of the error in $\|r_j\|^{-1}$ while CG allows a growth of the order of $\|r_j\|^{-2}\|p_j\|_A$ instead.

Adhoc approximations

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

- $\|E\|_{A^{-1},A} \geq \lambda_{\min}(A)^{-1}\|E\|_2$
- $\|p\|_A \approx \sqrt{\frac{1}{n}\mathrm{Tr}(A)}\|p\|_2$
  (ok for p with random independent components)
- $\|b\|_{A^{-1}} = \sqrt{2|q(x_*)|} \approx \sqrt{2|q_k|} \approx \sqrt{|b^T x_k|}$
- $\|H_k^{-1}\| = \frac{1}{\lambda_{\min}(H_k)} \leq \frac{1}{\lambda_{\min}(A)}$ (FOM only)
- $k_{\max} \approx \frac{\log(\epsilon)}{\log(\rho)}$ with $\rho \stackrel{\mathrm{def}}{=} \frac{\sqrt{\lambda_{\max}/\lambda_{\min}}-1}{\sqrt{\lambda_{\max}/\lambda_{\min}}+1}$

Termination test (Arioli & Gratton):

$$q_{k-d} - q_k \leq \tfrac{1}{4}\epsilon|q_k|$$

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

Does it still work (continuous accuracy levels)?

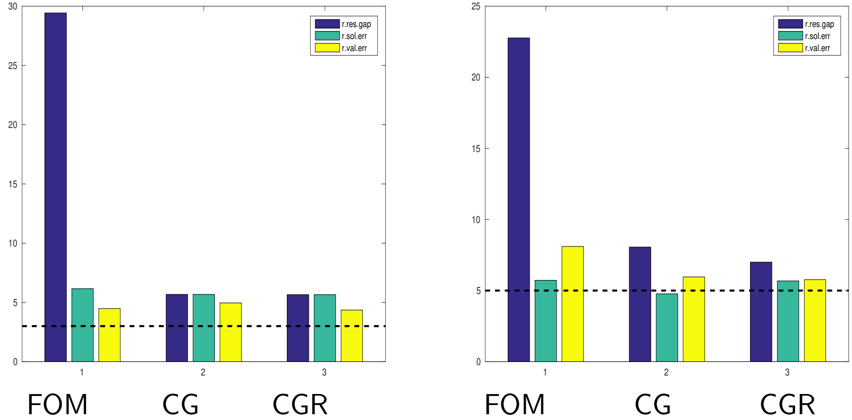


Figure: Exact bounds, $\kappa(A) = 10^1$, $\epsilon = 10^{-3}$ (left), $\kappa(A) = 10^5$, $\epsilon = 10^{-5}$ (right); continuous case

Does it still work (multiprecision)?

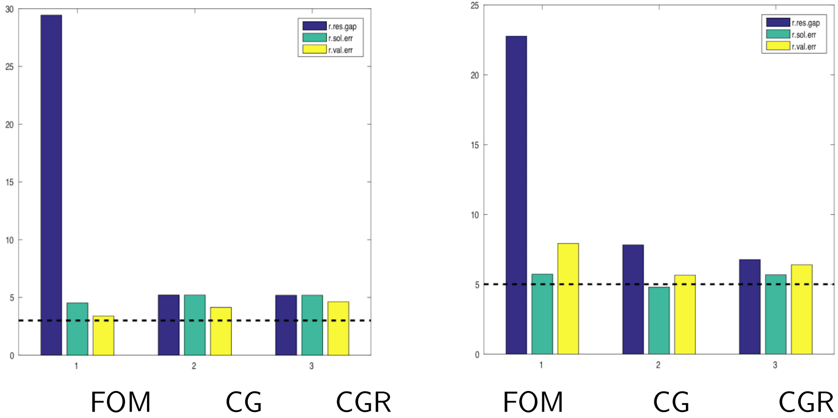


Figure: Approximate bounds, $\kappa(A) = 10^1$, $\epsilon = 10^{-3}$ (left), $\kappa(A) = 10^5$, $\epsilon = 10^{-5}$ (right); multiprecision

Outline for section 3

Consider

$$\min_{x\in\mathbf{R}^n} f(x).$$

The dynamic accuracy setting of trust-region methods [CGT 2000], it is assumed that

- The value of the objective can be approximated with a prespecified level of accuracy ω_f :
$$|\overline{f}(x,\omega_f) - \overline{f}(x,0)| \leq \omega_f \quad \text{and} \quad \overline{f}(x,0) = f(x)$$
- Following [Carter 1993; G., L.N Vicente and Z. Zhang 2020], the case where the gradient is inexact can be handled:
$$\|\overline{g}(x,\omega_g) - \overline{g}(x,0)\| \leq \omega_g \|\overline{g}(x,\omega_g)\| \;\text{ and }\; \overline{g}(x,0) = \nabla_x^1 f(x)$$

We recall that the convergence at step k

$$\|\nabla_x^1 f(x_k)\| \leq \|\overline{g}(x_k,\omega_{g,k})\| + \|\overline{g}(x_k,\omega_{g,k}) - \overline{g}(x,0)\| \leq \epsilon.$$

is gained provided, for some constant κ_g, $\omega_{g,k} \leq \kappa_g$ and $\|\overline{g}(x_k,\omega_{g,k})\| \leq \frac{\epsilon}{1+\kappa_g}$.

TR with dynamic accuracy on f and g (algo TR**1**DA) (Step computation)

Step 1 Check for termination. If $k = 0$ or $x_k \neq x_{k-1}$, choose $\omega_{g,k} \in (0, \kappa_g]$ and compute $\overline{g}_k = \overline{g}(x_k, \omega_{g,k})$ such that $\|\overline{g}(x_k, \omega_{g,k}) - \overline{g}(x_k, 0)\| \leq \omega_{g,k} \|\overline{g}(x, \omega_{g,k})\|$. Terminate if $\|\overline{g}(x_k, \omega_{g,k})\| \leq \frac{\epsilon}{1+\kappa_g}$.

Step 2 Step calculation. Sufficiently reduce the model $m(x_k, s) = f_k + \overline{g}_k^T s + \frac{1}{2} s^T H_k s$ in the Trust-Region $\{s_k, \|s_k\| \leq \Delta_k\}$ in the sense that

$$m(x_k, 0) - m(x_k, s_k) \geq \frac{1}{2} \|\overline{g}_k\| \min \left[\frac{\|\overline{g}_k\|}{\|H_k\|}, \Delta_k \right]$$

Step 3 Evaluate the objective function. Select $\omega_{f,k}^+ \in \Big(0, \eta_0 [m(x_k, 0) - m(x_k, s_k)] \Big]$ and compute $f_k^+ = \overline{f}(x_k + s_k, \omega_{f,k}^+)$. If $\omega_{f,k}^+ < \omega_{f,k}$, recompute $f_k = \overline{f}(x_k, \omega_{f,k}^+)$.

TR with dynamic accuracy on f and g (TR**1**DA) (Step acceptance)

Step 4 Acceptance of the trial point. Define the ratio

$$\rho_k = \frac{f_k - f_k^+}{m(x_k, 0) - m(x_k, s_k)}.$$

If $\rho_k \geq \eta_1$, then define $x_{k+1} = x_k + s_k$ and set $\omega_{f,k+1} = \omega_{f,k}^+$.
Otherwise set $x_{k+1} = x_k$, $\omega_{f,k+1} = \omega_{f,k}$ and $\omega_{g,k+1} = \omega_{g,k}$.

Step 5 Standard trust-radius update.

Set

$$\Delta_{k+1} \in \begin{cases} [\Delta_k, \infty) & \text{if } \rho_k \geq \eta_2, \quad \nearrow \\ [\gamma_2 \Delta_k, \Delta_k) & \text{if } \rho_k \in [\eta_1, \eta_2), \quad \searrow \\ [\gamma_1 \Delta_k, \gamma_2 \Delta_k] & \text{if } \rho_k < \eta_1. \quad \downarrow \end{cases}$$

Increment k by 1 and go to Step 2.

Assumptions

- AS.1: The objective function f is twice continuously differentiable in $\mathbb{R}^n$ and there exist a constant $\kappa_\nabla \geq 0$ such that $\|\nabla_x^2 f(x)\| \leq \kappa_\nabla$ for all $x \in \mathbb{R}^n$.
- AS.2: There exists a constant $\kappa_H \geq 0$ such that $\|H_k\| \leq \kappa_H$ for all $k \geq 0$.
- AS.3 There exists a constant κ_{low} such that $f(x) \geq \kappa_{\text{low}}$ for all $x \in \mathbb{R}^n$.

We can bound the accuracy on the model w.r.t the exact function:

Suppose AS.1 and AS.2 hold. Then, for each $k \geq 0$,

$$|f(x_k + s_k) - m(x_k, s_k)| \leq |f_k - f(x_k)| + \kappa_g \|\overline{g}(x_k, \omega_{g,k})\| \Delta_k + \kappa_{H\nabla} \Delta_k^2$$

for $\kappa_{H\nabla} = 1 + \max[\kappa_H, \kappa_\nabla]$.

The observed ρ can be interpreted as a true function versus model reduction

We have that, for all $k \geq 0$,

$$\max\left[|f_k - f(x_k)|, |f_k^+ - f(x_k+s_k)|\right] \leq \eta_0\left[m(x_k,0) - m(x_k,s_k)\right]$$

and

$$\rho_k \geq \eta_1 \quad \text{implies that} \quad \frac{f(x_k) - f(x_k+s_k)}{m(x_k,0) - m(x_k,s_k)} \geq \eta_1 - 2\eta_0 > 0.$$

Proof. This follows from the accuracy management and from

$$\rho_k = \frac{f_k - f_k^+}{m(x_k,0) - m(x_k,s_k)} = \frac{f(x_k) - f(x_k+s_k)}{m(x_k,0) - m(x_k,s_k)} + \frac{[f_k - f(x_k)] + \left[|f_k^+ - f(x_k+s_k)\right]}{m(x_k,0) - m(x_k,s_k)}$$

□

Suppose AS.1 and AS.2 hold, and that $\overline{g}(x_k, \omega_{g,k}) \neq 0$. Then

$$\Delta_k \leq \frac{\|\overline{g}(x_k, \omega_{g,k})\|}{2\kappa_{H\nabla}} \left[\tfrac{1}{2}(1-\eta_1)-\eta_0-\kappa_g\right] \text{ implies that } \Delta_{k+1} \geq \Delta_k.$$

Proof.

$$\begin{aligned}
|\rho_k - 1| &\leq \frac{|f_k^+ - f(x_k + s_k)| + |f(x_k + s_k) - m(x_k, s_k)|}{m(x_k, 0) - m(x_k, s_k)} \\
&\leq 2\eta_0 + \frac{\kappa_g \|\overline{g}(x_k, \omega_{g,k})\| \Delta_k + \kappa_{H\nabla} \Delta_k^2}{\frac{1}{2} \|\overline{g}(x_k, \omega_{g,k})\| \, \Delta_k} \\
&\leq 2\eta_0 + 2\kappa_g + 2\kappa_{H\nabla} \frac{\Delta_k}{\|\overline{g}(x_k, \omega_{g,k})\|} \\
&\leq 1 - \eta_2
\end{aligned}$$

where we used $\eta_0 + \kappa_g < \frac{1}{2}(1 - \eta_2)$. □

Suppose $\Delta_0 \geq \theta\epsilon$. The TR1DA algorithm produces an iterate x_k such that $\|\nabla_x^1 f(x_k)\| \leq \epsilon$ in at most $\tau_S \stackrel{\rm def}{=} \frac{2(f(x_0)-\kappa_{\rm low})(1+\kappa_g)}{(\eta_1-2\eta_0)\theta} \cdot \frac{1}{\epsilon^2}$ successful iterations, and at most

$$\tau_{\rm tot} \stackrel{\rm def}{=} \tau_S \left(1 - \frac{\log \gamma_3}{\log \gamma_2}\right) + \frac{1}{|\log \gamma_2|} \log \left(\frac{\Delta_0}{\theta\epsilon}\right) \qquad (3.3)$$

iterations in total.

Proof.

$$\begin{aligned}
f(x_0) - \kappa_{\rm low} &\geq \sum_{j\in\mathcal{S}_k} [f(x_j) - f(x_{j+1})] \\
&\geq \tfrac{1}{2}(\eta_1 - 2\eta_0) \sum_{j\in\mathcal{S}_k} \|\overline{g}(x_j,\omega_{g,j})\| \min\left[\frac{\|\overline{g}(x_j,\omega_{g,j})\|}{1+\|H_j\|}, \Delta_j\right] \\
&\geq \tfrac{1}{2}|\mathcal{S}_k|(\eta_1 - 2\eta_0)\frac{\epsilon}{1+\kappa_g} \min\left[\frac{\epsilon}{\kappa_{H\nabla}(1+\kappa_g)}, \min\left[\Delta_0, \theta\epsilon\right]\right] \\
&= |\mathcal{S}_k| \frac{(\eta_1 - 2\eta_0)}{2(1+\kappa_g)} \min\left[\frac{1}{\kappa_{H\nabla}(1+\kappa_g)}, \theta\right] \epsilon^2
\end{aligned}$$

Practical setting

In our numerical experiments with TR1DA

- We perfom 20 runs on 86 Cuter problems
- We assume that the objective function's value $\overline{f}(x_k, \omega_k)$ and the gradient $\overline{g}(x_k, \omega_k)$ can be computed with corresponding accuracy level equal to machine precision, half machine precision or quarter machine precision
- The computational cost of an operation is devided by 4 when passing from one level to the immediate next one: half precision corresponds to double-precision costs divided by 16
- Hessian approximation are obtained with a limited-memory symmetric rank-one (SR1) quasi-Newton update

Practical setting

To set the stage, our first experiment starts by comparing three variants of the TR1DA algorithm:

- LMQN: a version using $\omega_f = \omega_g = 0$ for all k (i.e. using the full double precision arithmetic throughout),
- LMQN-s: a version using single precision evaluation of the objective function and gradient for all k,
- LMQN-h: a version using half precision evaluation of the objective function and gradient for all k.

Simple minded approach: expensive parts of the optimization calculation conducted in reduced precision no further adaptive accuracy management.

Simple approach

						relative to LMQN		
ϵ	Variant	nsucc	its.	costf	costg	its.	costf	costg
1e-03	LMQN	82	41.05	42.04	42.04			
	LMQN-s	78	41.40	42.60	42.60	1.03	1.04	1.04
	LMQN-h	22	16.95	1.12	1.12	0.97	0.06	0.06
1e-05	LMQN	80	46.34	47.38	47.38			
	LMQN-s	48	47.79	48.96	48.96	1.08	1.08	1.08
	LMQN-h	10	17.80	1.18	1.18	1.38	0.08	0.08
1e-07	LMQN	67	62.76	63.85	63.85			
	LMQN-s	25	28.28	28.96	28.96	0.82	0.81	0.81
	LMQN-h	6	15.83	1.05	1.05	0.97	0.06	0.06

Table: Results for LMQN-s and LMQN-h compared to LMQN

- Quickly decreasing robustness when a tight accuracy is demanded
- In most cases, no improvement, in costf and costg
- When LMQN-h happens to succeed its cost is very low

Two variant of TR1DA

- LMQN: as above,
- iLMQN-a: a variant of the TR1DA algorithm where

$$\omega_{f,k} = \min[\tfrac{1}{10}, \tfrac{4}{100}\eta_1 \left(m_k(0) - m_k(s_k)\right)] \quad \text{and} \quad \omega_{g,k} = \tfrac{1}{2}\kappa_g.$$

- iLMQN-b: a variant of the TR1DA algorithm where,

$$\omega_{f,k} = \min[\tfrac{1}{10}, \tfrac{4}{100}\eta_1 \left(m_k(0) - m_k(s_k)\right)] \quad \text{and} \quad \omega_{g,k} = \min[\kappa_g, \omega_{f,k}].$$

Variable precision approach

						relative to LMQN		
ϵ	Variant	nsucc	its.	costf	costg	its.	costf	costg
1e-03	LMQN	82	41.05	42.04	42.04			
	iLMQN-a	80	50.05	9.88	6.11	1.23	0.24	0.15
	iLMQN-b	76	52.67	13.85	3.34	1.36	0.35	0.08
1e-05	LMQN	80	46.34	47.38	47.38			
	iLMQN-a	75	75.92	36.21	24.77	1.40	0.63	0.42
	iLMQN-b	63	72.57	39.85	4.60	1.78	0.95	0.11
1e-07	LMQN	67	62.76	63.85	63.85			
	iLMQN-a	47	65.83	58.97	37.50	1.18	1.03	0.65
	iLMQN-b	40	87.35	95.09	5.52	1.39	1.45	0.09

Table: Results for the variable-precision variants

Summary of the experiments

- For $\epsilon = 10^{-3}$ or 10^{-5}, inexact variants iLMQN-a and iLMQN-b perform well in cost for gradient and function
- iLMQN-a appears to dominate iLMQN-b in the evaluation of the objective function
- iLMQN-b shows significantly larger savings in the gradient evaluation costs
- When the final accuracy is tigther inexact methods appear to loose their edge in robustness. Gains in function evaluation costs disappear
- Comparison of iLMQN-a and even iLMQN-b with LMQN-s and LMQN-h clearly favours the new methods

Outline for section 4

Conclusions and perspectives

Summary:

- Optimization-focused theory to handle inexact function/gradient evaluation
- Theoretical gains substantial
- Translates well to practice after approximations

Perspectives:

- More general (controlable) inexactness in constrained optimization
- Probabilistic error specification

Thank your for your attention!

Reference

- S. Gratton, E. Simon, Ph. L. Toint,
  Minimizing convex quadratics with variable precision Krylov methods,
  arXiv:1807.07476, NLAA in press
- S. Gratton, Ph. L. Toint,
  A note on solving nonlinear optimization problems in variable precision,
  arXiv:1812.03467, COAP, 2020
- S. Gratton, E. Simon, Ph. L. Toint,
  Minimization of nonsmooth nonconvex functions using inexact evaluations and its worst-case complexity,
  arXiv:1807.07476, Mathematical Programming, 2020