Rapport de projet - phase 3

MTH8408

Oihan Cordelier, Oussama Mouhtal

Contexte et notations

On considère le système linéaire $Ax = b$.

Méthode des gradients conjugués (CG)

Hypothèse : A **HPD** (hermitienne définie positive).
Notations : x_k itéré, $r_k = b - Ax_k$ résidu, p_k direction de recherche.

Mises à jour (Saad, *Iterative Methods for Sparse Linear Systems*, chap. 6) :

$$\alpha_k = \frac{r_k^* r_k}{p_k^* A p_k}, \qquad x_{k+1} = x_k + \alpha_k p_k, \qquad r_{k+1} = r_k - \alpha_k A p_k,$$

$$\beta_{k+1} = \frac{r_{k+1}^* r_{k+1}}{r_k^* r_k}, \qquad p_{k+1} = r_{k+1} + \beta_{k+1} p_k.$$

Les directions $\{p_k\}$ sont A-conjuguées : $p_i^* A p_j = 0$ pour $i \neq j$.

DIOM / IOM (orthogonalisation incomplète)

IOM(m) construit une base $V_k = [v_1, \dots, v_k]$ de $\mathcal{K}_k(A, r_0)$ via une orthogonalisation **à fenêtre** de taille m (on orthogonalise v_{j+1} contre au plus m prédécesseurs).
DIOM est la variante « directe » qui met à jour x_k en se basant sur la projection $AV_k = V_k H_k + h_{k+1,k} v_{k+1}$ (Saad, *Iterative Methods for Sparse Linear Systems*, chap. 6).

Avec la factorisation $H_k = L_k U_k$, les **directions DIOM** se construisent par

$$p_k^{\text{diom}} = u_{kk}^{-1} \left(v_k - \sum_{i=k-m+1}^{k-1} u_{ik}\, p_i^{\text{diom}} \right),$$

où m est la taille de la mémoire et $u_{i,k}$ les entrées de U_k (pour $i \leq 0$, fixe $u_{ik}\, p_i^{\text{diom}} = 0$).
La mise à jour des itérées est donnée par : $x_k = x_{k-1} + \zeta_k\, p_k^{\text{diom}}$

Cas HPD. Lorsque (A) est HPD et que $m = 2$, IOM(2) reproduit **Lanczos** ; DIOM(2) coïncide alors, en arithmétique exacte, avec **FOM** (Galerkin) et donc avec **CG** (mêmes itérés) (Saad, *Iterative Methods for Sparse Linear Systems*, chap. 6).

Lien DIOM(2) – CG (arithmétique exacte)

Hypothèses : (A) HPD, (m=2).

- **Égalité des itérés** (Saad, *Iterative Methods for Sparse Linear Systems*, chap. 6):

$$x_k^{\text{cg}} = x_k^{\text{diom}} \qquad k = 0, 1, \dots, n.$$

- **Directions proportionnelles** (Saad, *Iterative Methods for Sparse Linear Systems*, chap. 6):

$$\alpha_k = 1/u_{k+1,k+1} \quad \text{et} \quad p_k^{\text{cg}} = \zeta_{k+1}\, u_{k+1,k+1}\, p_{k+1}^{\text{diom}}, \qquad k = 0, 1, \ldots, n \qquad (*).$$

Intuition : DIOM et CG imposent la même condition de Galerkin dans l'espace de Krylov engendré par Lanczos ; seules les **normalisations** des directions diffèrent.

L'algorithme `diom` avec région de confiance

Notre implémentation de `diom` avec région de confiance est dans le ficher `diom_tr.jl`. Dans cette section, nous expliquerons l'implémentation de `diom` avec région de confiance en détails et on fera des tests l'implémentation est inspier de l'implémentation de la région de confiance dans CG.

Région de confiance avec DIOM

- Dans l'algorithme `diom`, les itérés sont générés selon

$$x_k = x_{k-1} + \zeta_k\, p_k^{\text{diom}}.$$

  Utiliser p_k^{diom} pour chercher une racine σ de $\|x_{k-1} + \sigma\, p_k^{\text{diom}}\|^2 = \Delta^2$ est problématique, puisque il faut comparer σ à ζ_k qui n'a pas potentiellement un signe constant (La problèmeatique dans ce cas qu'elle racine faut choisir la plus grande ou la plus petite ou peut etre alterner suivant le signe de ζ_k??). Il est donc préférable pour le moment de passer par la direction de `cg`, calculée à partir de la relation $(*)$, ce qui nécessite de stocker un vecteur supplémentaire en mémoire.

- Il faut aussi calculer la norme de cet nouvelle direction (nommée dans l'algorithme `pcg`), ce qui correspond à une opération additionnelle. Lorsque $m = 2$ et que la matrice est hermitienne, on peut mettre à jour $\|p_k^{\text{diom}}\|$ comme dans `CG`. En revanche, dans le cas général, puisque

$$p_k^{\text{diom}} = \frac{1}{u_{kk}} \left(v_k - \sum_{i=k-m+1}^{k-1} u_{ik}\, p_i^{\text{diom}} \right),$$

  et que les vecteurs p_i ne sont pas orthogonaux entre eux, la mise à jour de $\|p_k^{\text{diom}}\|$ nécessite de connaître les produits $p_i^* p_j$. Autrement dit, il faudrait disposer de toutes les entrées de la matrice

$$P_k^* P_k \;=\; U_k^{-*} U_k^{-1} !!!$$

Test de la région de confaince dans `diom`

Les deux premiers tests sont inspirés des tests de `cg` dans `krylov`.
Le premier test consiste à vérifier que la solution est de norme nulle même si `radius > 0`.
Le deuxième test permet de s'assurer que, lorsque la courbure est négative, la variable `indefinite` est bien mise à jour.
Enfin, le troisième test repose sur la remarque que la façon dont la norme du résidu est calculée ne prend pas en compte que le coefficient $u_{k,k}$ est écrasé par $1/\sigma$.

```
using Test

function zero_rhs(n :: Int=10; FC=Float64)
  A = rand(FC, n, n)
  b = zeros(FC, n)
  return A, b
end
```

```
for FC in (Float64, ComplexF64)
  @testset "Data Type: $FC" begin
    # Test radius > 0  and b^TAb=0
    A, b = zero_rhs(FC=FC)
    solver = DiomTRWorkspace(A, b)
    diom!(solver, A, b,radius = 10 * real(one(FC)))
    x, stats = solver.x, solver.stats
    @test stats.status == "x is a zero-residual solution"
    @test norm(x) == zero(FC)
    @test stats.niter == 0

    # Test radius > 0 and p Ap < 0
    A = FC[
    10.0 0.0 0.0 0.0;
    0.0 8.0 0.0 0.0;
    0.0 0.0 5.0 0.0;
    0.0 0.0 0.0 -1.0
    ]
    b = FC[1.0, 1.0, 1.0, 0.1]
    solver = DiomTRWorkspace(A, b)
    diom!(solver, A, b; radius = 10 * real(one(FC)))
    x, stats = solver.x, solver.stats
    @test stats.indefinite == true

    # Test residus finale
    A = FC[
    10.0 0.0 0.0 0.0;
    0.0 8.0 0.0 0.0;
    0.0 0.0 5.0 0.0;
    0.0 0.0 0.0 -1.0
    ]
    b = FC[1.0, 1.0, 1.0, 0.1]
    solver = DiomTRWorkspace(A, b)
    diom!(solver, A, b; radius = 10 * real(one(FC)), history=true)
    x, stats = solver.x, solver.stats
    r_alg = stats.residuals[end]
    r_true = norm(b - A * x)
    rtol = sqrt(eps(real(one(FC))))
    @test !isapprox(r_true, r_alg; rtol = rtol)     # Les deux résidu se différent à tolérance machine
  end
end
```

```
Test Summary:      | Pass  Total  Time
Data Type: Float64 |    5      5  1.1s
Test Summary:         | Pass  Total  Time
Data Type: ComplexF64 |    5      5  1.1s
```

Proposition d'une méthode pour calculer $\|r_k\|$

Supposons qu'à une itération k, la normalisation de la direction dans `diom` est faite de la facon suivante:

$$\bar{p}_k^{\text{diom}} = \sigma \left(v_k - \sum_{i=k-m+1}^{k-1} u_{ik} \, p_i^{\text{diom}} \right).$$

où σ est racine de $\|x_{k-1} + \sigma\, p_k^{\text{diom}}\|^2 = \Delta^2$. Cette notation intermédiaire $\bar{p}_k^{\text{diom}}$ permet de réutiliser la vraie direction $p_k^{\text{diom}} = \frac{1}{u_{kk}}\left(v_k - \sum_{i=k-m+1}^{k-1} u_{ik}\, p_i^{\text{diom}}\right)$ par la suite.

L'itéré est donné par

$$x_{k+1} = x_k + \zeta_k \bar{p}_k^{\text{diom}}.$$

Le résidu r_k se calcule alors de la façon suivante :

$$r_k = r_{k-1} - \zeta_k A\bar{p}_k^{\text{diom}} = r_{k-1} - \zeta_k u_{k,k}\sigma A p_k^{\text{diom}}.$$

La matrice concaténant les directions de `diom` satisfait par définition :

$$P_k = V_k U_k^{-1}.$$

En multipliant par A, on obtient :

$$AP_k = AV_k U_k^{-1}.$$

Or, par la relation de récurrence d'Arnoldi :

$$AV_k = V_k H_k + h_{k+1,k} v_{k+1} e_k^\top.$$

Ainsi :

$$AP_k = (V_k H_k + h_{k+1,k} v_{k+1} e_k^\top)U_k^{-1} = (V_k L_k U_k + h_{k+1,k} v_{k+1} e_k^\top)U_k^{-1} = V_k L_k + h_{k+1,k} v_{k+1} e_k^\top U_k^{-1}.$$

En particulier,

$$Ap_k^{\text{diom}} = AP_k e_k = (V_k L_k + h_{k+1,k} v_{k+1} e_k^\top U_k^{-1})e_k = v_k + \frac{h_{k+1,k}}{u_{kk}} v_{k+1}.$$

Expression finale du résidu

On en déduit :

$$r_k = r_{k-1} - \zeta_k \sigma u_{k,k}\left(v_k + \frac{h_{k+1,k}}{u_{kk}} v_{k+1}\right).$$

En utilisant la relation (Dans le livre de Saad Chapitre 6)

$$r_{k-1} = -\zeta_{k-1}\frac{h_{k,k-1}}{u_{k-1,k-1}} v_k,$$

on obtient:

$$r_k = \left(-\zeta_{k-1}\frac{h_{k,k-1}}{u_{k-1,k-1}} - \zeta_k \sigma u_{k,k}\right) v_k - \zeta_k \sigma h_{k+1,k} v_{k+1}.$$

finalement utilsiant l'orthogonalité partielle des v_i, on obtient finalement :

$$\|r_k\| = \sqrt{\left|\zeta_{k-1}\frac{h_{k,k-1}}{u_{k-1,k-1}} + \zeta_k \sigma u_{k,k}\right|^2 + \left|\zeta_k \sigma h_{k+1,k}\right|^2}.$$

Reference

- Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd edition, SIAM, 2003.
- Y. Saad, *Practical use of some krylov subspace methods for solving indefinite and nonsymmetric linear systems*, SIAM journal on scientific and statistical computing, 5(1), pp. 203–228, 1984. " " "