Additional document

Conjugate gradient algorithm to solve the primal interface problem

In practice, the interface problem is never assembled to be solved by a direct solver. The use of an iterative solver is much less expensive and more efficient since it only involves the product of local matrix-vector products which can be done in parallel in a distributed way.

In this document, the application of the conjugate gradient algorithm is proposed in order to solve the primal interface problem :

$$\mathbf{S}_p \mathbf{u}_b = \mathbf{b}_p$$

with :

$$\begin{aligned} \mathbf{S}_p &= \mathbb{A}^{\text{⟠}} \, S_p^{\text{◈}} \, \mathbb{A}^{\text{⟠}\,T} \\ \mathbf{b}_p &= \mathbb{A}^{\text{⟠}} \, b_p^{\Diamond} \end{aligned}$$

1 Conjugate gradient algorithm

The conjugate gradient algorithm to solve a linear system $Ax = b$ is given in **Algorithm 1**.

Algorithm 1 Conjugate gradient

```
**Initialization :** Choose $x_0$ (generally $x_0 = 0$)
□ $r_{(0)} = b - A x_{(0)}$
□ $d_{(0)} = r_{(0)}$
**for** For $i = 0$ to $m$ **do**
    1. Compute optimal step : $\alpha_{(i)} = \dfrac{r_{(i)}^T r_{(i)}}{d_{(i)}^T A d_{(i)}} \left( = \dfrac{r_{(i)}^T d_{(i)}}{d_{(i)}^T A d_{(i)}} \right)$
    2. Compute iterate : $x_{(i+1)} = x_{(i)} + \alpha_{(i)} d_{(i)}$
    3. Compute residual : $r_{(i+1)} = r_{(i)} - \alpha_{(i)} A d_{(i)}$
    4. Compute orthogonalization parameter : $\beta_{(i+1)} = \dfrac{r_{(i+1)}^T r_{(i+1)}}{r_{(i)}^T r_{(i)}} \left( = -\dfrac{r_{(i+1)}^T A d_{(i)}}{d_{(i)}^T A d_{(i)}} \right)$
    5. Update search direction : $d_{(i+1)} = r_{(i+1)} + \beta_{(i+1)} d_{(i)}$
**end for**
```

2 Conjugate gradient algorithm to solve the primal interface problem

By considering **Algorithm 1** to solve a linear system $Ax = b$, one can see that matrix-vector product Ad is required. When applied to solve the primal interface problem, computing matrix-vector product $\mathbf{S}_p \mathbf{r}_b$ can be done thanks to local (subdomain) matrix-vector products. One recall that bold characters refer to assembled quantities.

Let consider the initialization step of the algorithm. Knowing the initialization $\mathbf{u}_{b(0)}$, one deduces that for each subdomain :

$$\left\{ \quad \forall\, s \in \mathbf{S}, \quad \begin{pmatrix} K_{ii}^{(s)} & K_{ib}^{(s)} \\ K_{bi}^{(s)} & K_{bb}^{(s)} \end{pmatrix} \begin{pmatrix} u_{i(0)}^{(s)} \\ u_{b(0)}^{(s)} \end{pmatrix} \quad = \quad \begin{pmatrix} f_i^{(s)} \\ f_b^{(s)} \end{pmatrix} \right.$$

where $u_{b(0)}{}^{\diamondsuit} = \mathbb{A}^{\Diamond}{}^{T} \mathbf{u}_{b(0)}$ is known. In this case, from the first line of the previous system, one can obtain the internal unknowns :

$$u_{i(0)}^{(s)} = K_{ii}^{(s)^{-1}} \left(f_i^{(s)} - K_{ib}^{(s)} u_{b(0)}^{(s)} \right)$$

which corresponds to the solution of a Dirichlet problem with prescribed displacement $u_{b(0)}^{(s)}$. One deduces that :

$$\left\{ \quad \forall\, s \in \mathbf{S}, \quad \begin{pmatrix} K_{ii}^{(s)} & K_{ib}^{(s)} \\ K_{bi}^{(s)} & K_{bb}^{(s)} \end{pmatrix} \begin{pmatrix} u_{i(0)}^{(s)} \\ u_{b(0)}^{(s)} \end{pmatrix} - \begin{pmatrix} f_i^{(s)} \\ f_b^{(s)} \end{pmatrix} = \begin{pmatrix} 0 \\ -r_{b(0)}^{(s)} \end{pmatrix} \quad = \quad \begin{pmatrix} 0 \\ S_p^{(s)} u_{b(0)}^{(s)} - b_p^{(s)} \end{pmatrix} \right.$$

The second line of the system reflects the equilibrium residual of the subdomain and represents exactly the contribution $r_{b(0)}^{(s)} = b_p^{(s)} - S_p^{(s)} u_{b(0)}^{(s)}$ to the assembled residual $\mathbf{r}_{b(0)} = \mathbf{b}_p - \mathbf{S}_p \mathbf{u}_{b(0)}$. Thus, the assembled residual can be subsequently computed as follows :

$$\mathbf{r}_{b(0)} = \mathbb{A}^{\Diamond}\, r_{b(0)}{}^{\diamondsuit}$$

It is worth noting at the fact that it is consequently not necessary to compute explicitly the Schur complement $S_p^{(s)}$. Only the computation of the substructure equilibrium residual $r_{b(0)}^{(s)}$ is required. One can also easily check that one finally computes :

$$\mathbf{r}_{b(0)} = \mathbb{A}^{\Diamond}\, r_{b(0)}{}^{\diamondsuit} = \mathbb{A}^{\Diamond} \left(b_p{}^{\diamondsuit} - S_p{}^{\lozenge}\, u_b{}^{\diamondsuit} \right) = \mathbf{b}_p - (\mathbb{A}^{\Diamond}\, S_p{}^{\lozenge}\, \mathbb{A}^{\Diamond}{}^{T}) \mathbf{u}_{b(0)} = \mathbf{b}_p - \mathbf{S}_p \mathbf{u}_{b(0)}$$

which represents the residual of the primal interface problem. Initial search direction is defined as $\mathbf{d}_{b(0)} = \mathbf{r}_{b(0)}$.

Let now consider a given iteration $k+1$. Quantities $\mathbf{u}_{b(k)}$, $\mathbf{d}_{b(k)}$ and $\mathbf{r}_{b(k)}$ at previous iteration k are known. Following the same procedure as before, one can compute the product by the Schur complement of any vector $\mathbf{d}_b$ thanks to local (subdomain) computations as follows :

- Compute local (concatenated) vector : $d_{b(k)}{}^{\diamondsuit} = \mathbb{A}^{\Diamond}{}^{T} \mathbf{d}_{b(k)}$
- Solve local Dirichlet problem : $\forall\, s \in \mathbf{S}$, $d_{i(k)}^{(s)} = -K_{ii}^{(s)^{-1}} K_{ib}^{(s)} d_{b(k)}^{(s)}$
- Compute local matrix-vector product :

$$\left\{ \quad \forall\, s \in \mathbf{S}, \quad \begin{pmatrix} K_{ii}^{(s)} & K_{ib}^{(s)} \\ K_{bi}^{(s)} & K_{bb}^{(s)} \end{pmatrix} \begin{pmatrix} d_{i(k)}^{(s)} \\ d_{b(k)}^{(s)} \end{pmatrix} \quad = \quad \begin{pmatrix} 0 \\ S_p^{(s)} d_{b(k)}^{(s)} \end{pmatrix} \right.$$

- Assemble and compute global matrix-vector product : $\mathbf{S}_p \mathbf{d}_{b(k)} = \mathbb{A}^{\Diamond} \left(S_p{}^{\lozenge}\, d_{b(k)}{}^{\diamondsuit} \right)$

Once again, one can note that only local vectors $S_p^{(s)} d_{b(k)}^{(s)}$ have to be computed and exchanged between subdomains/proessors which represents a small amount of data. The local primal Schur complement $S_p^{(s)}$ does not have to be computed or exchanged between subdomains/processors. Once matrix-vector product $\mathbf{S}_p \mathbf{d}_{b(k)}$ is known, one can proceed to next steps of the the conjugate gradient algorithm. This leads to **Algorithm 2**.

Remark 1 *New iterate is computed as follows :*

$$\mathbf{u}_{b(k+1)} = \mathbf{u}_{b(k)} + \alpha_{(k)} \mathbf{d}_{b(k)} \quad \Rightarrow \quad u_{b(k+1)}{}^{\diamondsuit} = \mathbb{A}^{\Diamond}{}^{T} \mathbf{u}_{b(k+1)}$$

Internal degrees of freedom within subdomains can be easily updated if needed :

$$\begin{aligned} K_{ii}^{(s)} u_{i(k+1)}^{(s)} &= f_i^{(s)} - K_{ib}^{(s)} u_{b(k+1)}^{(s)} \\ &= f_i^{(s)} - K_{ib}^{(s)} \left(u_{b(k)}^{(s)} + \alpha_{(k)} d_{b(k)}^{(s)} \right) \end{aligned}$$

Thus,

$$\begin{aligned} u_{i(k+1)}^{(s)} &= K_{ii}^{(s)^{-1}} \left(f_i^{(s)} - K_{ib}^{(s)} u_{b(k)}^{(s)} \right) + \alpha_{(k)} \left(-K_{ii}^{(s)^{-1}} d_{b(k)}^{(s)} \right) \\ &= u_{i(k)}^{(s)} + \alpha_{(k)} d_{i(k)}^{(s)} \end{aligned}$$

where quantities $u_{i(k)}^{(s)}$ and $d_{i(k)}^{(s)}$ have already been computed.

Remark 2 *Residual of the initial (unsubstructured) problem is directly linked to the residual of the interface problem :*

$$\begin{aligned}\mathbf{r}_{(k)} &= \boldsymbol{f} - \boldsymbol{K}\boldsymbol{u}_{(k)} = \sum_s \left[\begin{pmatrix} f_i^{(s)} \\ f_b^{(s)} \end{pmatrix} - \begin{pmatrix} K_{ii}^{(s)} & K_{ib}^{(s)} \\ K_{bi}^{(s)} & K_{bb}^{(s)} \end{pmatrix}\begin{pmatrix} u_{i(0)}^{(s)} \\ u_{b(0)}^{(s)} \end{pmatrix}\right] \\ &= \sum_s \begin{pmatrix} 0 \\ b_p^{(s)} - S_p^{(s)} u_b^{(s)} \end{pmatrix} = \sum_s \begin{pmatrix} 0 \\ r_{b(0)}^{(s)} \end{pmatrix}\end{aligned}$$

Thus : $\|\mathbf{r}_{(k)}\| = \|\boldsymbol{f} - \boldsymbol{K}\boldsymbol{u}_{(k)}\| = \|\mathbf{r}_{b(k)}\|_\Gamma = \|\mathbf{b}_p - \mathbf{S}_p\mathbf{u}_{b(k)}\|_\Gamma$

Algorithm 2 Conjugate gradient algorithm to solve the primal interface problem

```
Initialization : Choose $\mathbf{u}_{b(0)}$ (generally $\mathbf{u}_{b(0)} = 0$)
□ Compute local (concatenated) vector : $u_{b(0)}^{\text{⟐}} = \mathbb{A}^{\text{⟠}^T} \mathbf{u}_{b(0)}$
for For $s = 1$ to $N_s$ do
  1.   Solve local Dirichlet problem : $u_{i(0)}^{(s)} = K_{ii}^{(s)^{-1}} \left(f_i^{(s)} - K_{ib}^{(s)} u_{b(0)}^{(s)}\right)$
  2.   Compute local matrix-vector product :
       $$\begin{pmatrix} K_{ii}^{(s)} & K_{ib}^{(s)} \\ K_{bi}^{(s)} & K_{bb}^{(s)} \end{pmatrix}\begin{pmatrix} u_{i(0)}^{(s)} \\ u_{b(0)}^{(s)} \end{pmatrix} - \begin{pmatrix} f_i^{(s)} \\ f_b^{(s)} \end{pmatrix} = \begin{pmatrix} 0 \\ -r_{b(0)}^{(s)} \end{pmatrix} = \begin{pmatrix} 0 \\ S_p^{(s)} u_{b(0)}^{(s)} - b_p^{(s)} \end{pmatrix}$$
end for
□ Compute global residual : $\mathbf{r}_{b(0)} = \mathbb{A}^{\text{⟠}} r_{b(0)}^{\text{⟐}} \left(= \mathbf{b}_p - \mathbf{S}_p\mathbf{u}_{b(0)}\right)$
□ $\mathbf{d}_{b(0)} = \mathbf{r}_{b(0)}$
for For $k = 0$ to $m$ do
  □   Compute local (concatenated) vector : $d_{b(k)}^{\text{⟐}} = \mathbb{A}^{\text{⟠}^T} \mathbf{d}_{b(k)}$
    for For $s = 1$ to $N_s$ do
—        Solve local Dirichlet problem : $\forall s \in \mathbf{S},\ d_{i(k)}^{(s)} = -K_{ii}^{(s)^{-1}} K_{ib}^{(s)} d_{b(k)}^{(s)}$
—        Compute local matrix-vector product :
         $$\left\{ \forall s \in \mathbf{S}, \quad \begin{pmatrix} K_{ii}^{(s)} & K_{ib}^{(s)} \\ K_{bi}^{(s)} & K_{bb}^{(s)} \end{pmatrix}\begin{pmatrix} d_{i(k)}^{(s)} \\ d_{b(k)}^{(s)} \end{pmatrix} = \begin{pmatrix} 0 \\ S_p^{(s)} d_{b(k)}^{(s)} \end{pmatrix} \right.$$
    end for
  □   Compute global matrix-vector product : $\mathbf{S}_p\mathbf{d}_{b(k)} = \mathbb{A}^{\text{⟠}} \left(S_p^{\Diamond}\ d_{b(k)}^{\text{⟐}}\right)$
  □   Compute optimal step : $\alpha_{(k)} = \dfrac{\mathbf{r}_{b(k)}^T \mathbf{r}_{b(k)}}{\mathbf{d}_{b(k)}^T \mathbf{S}_p \mathbf{d}_{b(k)}} \left(= \dfrac{\mathbf{r}_{b(k)}^T \mathbf{d}_{b(k)}}{\mathbf{d}_{b(k)}^T \mathbf{S}_p \mathbf{d}_{b(k)}}\right)$
  □   Compute iterate : $\mathbf{u}_{b(k+1)} = \mathbf{u}_{b(k)} + \alpha_{(k)}\mathbf{d}_{b(k)}$
  □   Compute residual : $\mathbf{r}_{b(k+1)} = \mathbf{r}_{b(k)} - \alpha_{(k)}\mathbf{S}_p\mathbf{d}_{b(k)}$
  □   Compute orthogonalization parameter :
      $$\beta_{(k+1)} = \frac{\mathbf{r}_{b(k+1)}^T \mathbf{r}_{b(k+1)}}{\mathbf{r}_{b(k)}^T \mathbf{r}_{b(k}} \left(= -\frac{\mathbf{r}_{b(k+1)}^T \mathbf{S}_p \mathbf{d}_{b(k)}}{\mathbf{d}_{b(k)}^T \mathbf{S}_p \mathbf{d}_{b(k)}}\right)$$
—     Update search direction : $\mathbf{d}_{b(k+1)} = \mathbf{r}_{b(k+1)} + \beta_{(k+1)}\mathbf{d}_{b(k)}$
end for
```