반복-직접 희소 솔버 조합에 의한 대규모 유한요소 모델의 주파수 영역 해석의 계산 효율

조 정 래¹ · 조 근 희^{1*}

¹한국건설기술연구원 인프라안전연구본부

Computational Efficiency on Frequency Domain Analysis of Large-scale Finite Element Model by Combination of Iterative and Direct Sparse Solver

Jeong-Rae Cho¹ and Keunhee Cho^{1*}

¹Department of Infrastructure Safety Research, Korea Institute of Civil Engineering and Building Technology, Goyang, 10223, Korea

Abstract

Parallel sparse solvers are essential for solving large-scale finite element models. This paper introduces the combination of iterative and direct solver that can be applied efficiently to problems that require continuous solution for a subtly changing sequence of systems of equations. The iterative-direct sparse solver combination technique, proposed and implemented in the parallel sparse solver package, PARDISO, means that iterative sparse solver is applied for the newly updated linear system, but it uses the direct sparse solver's factorization of previous system matrix as a preconditioner. If the solution does not converge until the preset iterations, the solution will be sought by the direct sparse solver, and the last factorization results will be used as a preconditioner for subsequent updated system of equations. In this study, an improved method that sets the maximum number of iterations dynamically at the first Krylov iteration step is proposed and verified thereby enhancing calculation efficiency by the frequency domain analysis.

Keywords : large-scale finite element model, frequency domain analysis, parallel sparse solver, combination of iterative and direct sparse solver

1. 서 론

대규모 유한요소 모델을 빠르게 해석하기 위해서는 컴퓨터 하드웨어 및 소프트웨어의 발전을 고려한 유한요소해석 프로그램의 설계가 필요하다. 특히 최적화된 수치라이브러리의 사용, 최신 희소솔버의 도입 등을 통해 그 효율성을 높일 수 있다 (Cho, et al., 2017). 이 논문에서는 미세하게 변화하는 시스템 행렬을 대상으로 연속적으로 해를 구해야 하는 주파수 해석 문제에서 효율적으로 적용 가능한 반복-직접 희소 솔버 조합 기법을 소개한다.

유한요소해석 프로그램은 어셈블된 선형시스템을 대상으로

해를 구하는 희소 솔버 적용을 통해 계산 시간을 대폭 단축할 수 있다. 희소 솔버는 LU 분해에 근거한 직접 희소 솔버와 Krylov법에 근거한 반복 희소 솔버로 구분할 수 있다(Sadd, 2003; Davis, 2006). 직접 희소 솔버는 한번 행렬 분해 (factorization)을 수행하고 나면 여러 번 우변 벡터에 적용 가능하며, 행렬의 상태가 나쁘더라도 동일한 시간이 소요되는 등의 장점을 지니고 있다. 하지만 필인(fill-in)의 존재로 메모리가 많이 소요되는 단점을 가지고 있다. 반면에 반복 희소 솔버는 메모리 요구량이 작고 문제 유형에 따라 빠르게 수렴하는 장점을 지니나 동일한 행렬 구조를 갖더라도 행렬값에 따라 행렬 상태가 나쁠 경우 수렴하지 않거나 반복회수가 늘어나는 단점이

* Corresponding author:

Tel: +82-31-910-0132; E-mail: kcho@kict.re.kr

Received December 21 2018; Revised January 2 2019;

Accepted January 3 2019

©2019 by Computational Structural Engineering Institute of Korea

This is an Open-Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License(<http://creativecommons.org/licenses/by-nc/3.0>) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

있다(Cho, *et al.*, 2017).

본 연구에서는 반복-직접 희소 솔버 조합 기법을 소개한다. 이 기법은 병렬 희소 솔버 패키지인 PARDISO에 제안 및 구현된 기법으로, 새롭게 행렬값이 갱신된 선형 시스템의 해를 구할 때 이전 선형 시스템에 적용된 직접 희소 솔버의 행렬 분해(factorization) 결과를 Krylov 반복 희소 솔버의 preconditioner로 활용하는 방법을 의미한다. PARDISO에서는 미리 설정된 반복 회수까지 해가 수렴하지 않으면 직접 희소 솔버로 해를 구하며, 이후 이어지는 갱신된 선형 시스템의 해를 구할 때는 최종적으로 사용된 직접 희소 솔버의 행렬 분해 결과를 preconditioner로 사용한다. 이 연구에서는 첫 번째 Krylov 반복 단계에서 소요되는 시간을 동적으로 계산하여 최대 반복 회수를 설정하는 기법을 제안하였으며, 주파수 영역 해석에 적용하여 그 효과를 검증하였다.

2. 반복-직접 희소 솔버 조합 기법

반복-직접 희소 솔버 조합 기법은 병렬 희소 솔버 패키지인 PARDISO에 제안 및 구현된 기법이다. 새롭게 행렬값이 갱신된 선형 시스템의 해를 구할 때 이전 선형 시스템에 적용된 직접 희소 솔버의 행렬 분해(factorization) 결과를 Krylov 반복 희소 솔버의 preconditioner로 활용하는 방법으로, 반복 솔버와 직접 솔버의 장점을 선택으로 적용하여 계산 효율을 올리게 된다. 이 기법을 설명을 위해 연속적으로 풀어야 하는 두 개의 시스템 $A^1x^1 = b^1$, $A^2x^2 = b^2$ 을 가정하기로 한다.

첫 번째 시스템 $A^1x^1 = b^1$ 에는 Fig. 1과 같은 직접 솔버 알고리즘을 적용한다. 직접 솔버 알고리즘은 세 단계로 구성되며, 분석 단계(analysis phase)는 행렬의 희소 패턴을 분석하여 수식번호를 최적화하는 단계이다. 분해 단계(factorization)은 행렬을 삼각행렬인 L과 U로 분해하는 과정이다. 솔루션 단계(solution phase)는 분해된 삼각행렬로 해를 구하는 단계이다. 만약 행렬의 희소 패턴이 변화하지 않는 경우 분석 단계는 1회 계산만 필요하다. 계산 시간은 대부분 분해 단계에서 소요되며, 이미 분해가 이루어진 경우에는 우변 벡터가 새롭게 주어지더라도 솔루션 단계는 빠르게 계산 가능하다.

두 번째 시스템 $A^2x^2 = b^2$ 에는 Krylov 반복 희소 솔버 알고리즘을 적용하는데, 이 때 preconditioner로 이전 시스템 행렬 A_1 인 활용한다. 만약 수렴하지 않는다면 직접 희소 솔버를 적용

Given $Ax = b$, M

1. Compute $r_0 := b - Ax_0$ for some initial guess x_0
2. Choose $\tilde{r}_0 := \text{conj}(r_0)$ or random
3. for $j = 1, 2, \dots$
4. $\rho_{j-1} := (\tilde{r}_{j-1}, r_{j-1})$
5. if $j = 1$
6. $u_1 := r_0$
7. $p_1 := u_1$
8. else
9. $\beta_{j-1} = \rho_{j-1} / \rho_{j-2}$
10. $u_j := r_{j-1} + \beta_{j-1}q_{j-1}$
11. $p_j = u_j + \beta_{j-1}(q_{j-1} + \beta_{j-1}p_{j-1})$
12. endif
13. Solve $M\hat{p} = p_j$
14. $\hat{v} := A\hat{p}$
15. $\alpha_j := \rho_{j-1} / (\tilde{r}_{j-1}, \hat{v})$
16. $q_j := u_j - \alpha_j\hat{v}$
17. Solve $M\hat{u} = u_j + q_j$
18. $x_j := x_{j-1} + \alpha_j\hat{u}$
19. $\hat{q} := A\hat{u}$
20. $r_j := r_{j-1} - \alpha_j\hat{q}$
21. Check convergency, continue if necessary
22. endfor

Note that $\text{conj}(x) = \bar{x}$, $(x, y) = x^H y$

Fig. 2 Preconditioned CGS algorithm

한다.

$Ax = b$ 형태의 선형시스템을 풀 때 반복 솔버는 Krylov 알고리즘을 적용하는데 해의 수렴성을 높이기 위해 행렬 A 에 근접하도록 preconditioner M 을 도입하여 $(M \approx A)$ $M^{-1}Ax = M^{-1}b$ 의 해를 구하게 된다. Preconditioner로는 Jacobi, SSOR, ILU 등이 주로 사용된다. Fig. 2는 preconditioner를 활용한 Conjugate Gradient Squared(이하 CGS) 알고리즘을 나타낸 것이다(Bai, *et al.*, 2002; LIS solver). Positive definite 행렬에는 Conjugate Gradient(이하 CG), non-positive definite 행렬에는 소개한 CGS 외에도 BiConjugate Gradient(이하 BiCG), BiConjugate Gradient Stabilized(이하 BiCGStab) Generalized Minimum Residual(이하 GMRES) 등의 알고리즘을 적용할 수 있다. 반복 솔버

Given $Ax = y$

1. Analysis phase : Compute P from A's sparsity for minimizing fill-in
2. Factorization phase : Compute $LU = PA$
3. Solution phase : $LUx = Py$

Fig. 1 Solver phases of direct sparse solver(Cho, *et al.*, 2017)

의 수렴성 검토는 일반적으로 $\|r_j\|/\|b\| < \epsilon$ 를 부과하며, 여기에서 ϵ 은 수렴 허용치로 $10^{-5} \sim 10^{-6}$ 이다.

반복-직접 회소 솔버 조합 기법에서는 $A^2x^2 = b^2$ 의 해를 구할 때 $(A^1)^{-1}A^2x^2 = (A^1)^{-1}b^2$ 의 해를 구하게 되며, 행렬 A^1 과 A^2 가 근접하고 있다면 작은 반복 단계만으로 해를 구할 수 있다. Fig. 2의 알고리즘을 살펴보면 $Mz_{j-1} = r_{j-1}$ 등과 같이 preconditioner M 으로 구성된 시스템의 해를 구하는 단계가 있다. 이 시스템에서 M 은 이미 직접 솔버 알고리즘을 통해 LU 분해가 수행되어 있기 때문에 빠르게 계산이 가능하다.

PARDISO에서 제시하는 반복-직접 회소 솔버 조합 기법은 이미 직접 솔버 알고리즘으로 행렬 분해된 결과가 존재하는 경우 갱신된 행렬 시스템에 대해 CG/CGS 반복 단계를 적용하는데, 최대 반복 회수를 150회로 고정하고 있다. 만약 150회 이내에 수렴하지 않으면 직접 솔버 알고리즘을 적용하도록 구성되어 있다. 직접 솔버의 행렬 분해에 소요되는 시간이 CG/CGS 반복을 150회를 수행하는 시간의 2배라는 예측에서 반복회수 150회가 결정되었다고 보고하고 있다. 본 연구에서는 이와 같이 고정된 최대 반복 회수를 사용하는 알고리즘을 CombinationS라 지칭하며, Fig. 3는 이 알고리즘을 요약한 것이다. PARDISO에서는 Krylov iteration으로 positive definite 행렬에 대해서는 CG, indefinite 행렬에 대해서는 CGS를 적용한다.

CombinationS 방법의 단점은 Krylov 반복에 대한 최대 반복 회수(maximum number of krylov iteration)를 미리 설정한다는 점이다. PARDISO에서 사용하는 CG/CGS 반복 최대 회수 150회는 직접 솔버의 행렬 분해에 소요되는 시간이 CG/CGS 반복을 150회를 수행하는 시간의 2배라는 예측에서 설정되었지만, 적용하는 문제, 계산에 참여하는 코어수, 직접 회소 솔버의 알고리즘의 병렬화 정도에 따라 이 예측을 틀릴 수 있다. 예를 들어 특정 조건에서 직접 솔버의 행렬 분해에 소요되는 시간이 100초이고 CG/CGS 반복 단계를 1회 수행하는 시간이 10초라고 한다면, CG/CGS 반복 단계가 10회 이내에 수렴해야 반복-직접 회소 솔버 조합 기법이 효율적인 방법이 되기 때문이다. 따라서 적용 문제, 계산에 참여하는 CPU 코어 수, 직접 솔버 알고리즘과 반복 솔버 알고리즘의 병렬화 정도 등에 따라 최대 반복 회수를 실행시간(run-time)에서 동적으로 결정할 필요가 있다. 본 연구에서는 첫 번째 Krylov 반복단계에서 행렬 분해(factorization)에 소요되는 시간 T_F 와 Krylove 반복 단계 1회에 소요되는 시간 T_K 를 비교하여 최대반복회수 n_{max} 를 계산하는 CombinationD 알고리즘은 제안하였다(Fig. 4). CombinationD 알고리즘은 최대반복회수를 런타임(run-time)에서 결정하기 때문에 수렴성이 나쁜 문제에서 과도하게 Krylov 반복 단계를 수행하는

```

Given  $Ax=b$ 
1. if  $M$  is none,
2.    $M:=A$  and factorize  $M=LU$ 
3. endif
4. Compute  $r_0 := b - Ax_0$  for some intial guess  $x_0$ 
5.  $j := 1$ 
6. while(true)
7.   ...
8.   Krylov iteration
9.   ...
10.  if  $j = n_{max}$ , break(failed)
11.   $j := j + 1$ 
12. endwhile
13. if failed
14.    $M:=A$  and factorize  $M=LU$ 
15.   Solve  $Mx=b$ 
16. endif

```

Fig. 3 Algorithm of combination of iterative and direct sparse solver with static iteration number (CombinationS)

```

Given  $Ax=b$ 
1. if  $M$  is none,
2.    $M:=A$  and factorize  $M=LU$ 
3. endif
4. Compute  $r_0 := b - Ax_0$  for some intial guess  $x_0$ 
5.  $j := 1$ 
6. while(true)
7.   ...
8.   Krylov iteration
9.   ...
10.  if  $j = 1$ 
11.     $n_{max} := T_F / T_K$ 
12.  endif
13.  if  $j = n_{max}$ , break(failed)
14.   $j := j + 1$ 
15. endwhile
16. if failed
17.    $M:=A$  and factorize  $M=LU$ 
18.   Solve  $Mx=b$ 
19. endif

```

Fig. 4 Algorithm of combination of iterative and direct sparse solver with dynamic maximum iteration number(CombinationD)

것을 막아주고, 반대로 과도하게 행렬 분해에 소요되는 시간이 요구되는 경우 충분한 Krylov 반복 단계를 수행함으로써 계산의 효율성을 높일 수 있다.

3. 수치 예제

본 연구에서 제안한 방법에 대한 효과를 검토하기 위해 미세

Fig. 5 Cantilever model

Fig. 6 Discretization of model

하게 변화하는 시스템 행렬을 갖는 문제인 주파수 영역 해석을 수행하였다. Fig. 5와 같은 두 유형의 캔틸레버를 대상으로 Fig. 6과 같이 8절점 솔리드 요소를 x , y , z 방향으로 n_x , n_y , n_z 개로 분할하는 해석 모델 8개를 고안하였다. Table 1은

각 해석 모델의 방향별 분할 개수, 요소수, 절점 수, 자유도 수, half bandwidth 등을 정리한 것이다. L-x 모델은 부재축 방향 길이가 긴 모델이며, 반면에 S-x는 짧은 모델이다. x 는 근사적인 자유도의 개수를 나타내는데, 2는 약 200만 자유도,

Table 1 Cantilever analysis model cases

Model	NX	NY	NZ	No. of Elements	No. of Nodes	No. of DOFs	Half. band width
L-2	500	50	25	625,000	664,326	1,992,978	4056
S-2	320	64	32	655,360	688,545	2,065,635	6534
L-1	400	40	20	320,000	345,261	1,035,783	2646
S-1	250	50	25	312,500	332,826	998,478	4056
L-0.4	300	30	15	135,000	149,296	447,888	1536
S-0.4	190	38	19	137,180	148,980	446,940	2400
L-0.2	240	24	12	69,120	78,325	234,975	1014
S-0.2	150	30	15	67,500	74,896	224,688	1536

Fig. 7 Analysis model

1은 약 100만 자유도, 0.2는 20만 자유도, 0.1은 10만 자유도 등이다. S-x가 L-x에 비해 비슷한 개수의 자유도에 대해 half-band width가 더 크다.

캔틸레버 해석 모델을 대상으로 Fig. 7과 같은 지반 운동을 가정한 주파수 영역 해석을 수행하였다. 선형 동적계의 운동 방정식은 다음과 같다.

$$\mathbf{M}\ddot{\mathbf{u}}(t) + \mathbf{C}\dot{\mathbf{u}}(t) + \mathbf{K}\mathbf{u}(t) = \mathbf{f}(t) \quad (1)$$

여기에서 $\mathbf{M}$, $\mathbf{C}$, $\mathbf{K}$ 는 각각 질량 행렬, 감쇠 행렬, 강성 행렬이다. $\mathbf{u}(t)$ 와 $\mathbf{f}(t)$ 는 각각 변위 벡터와 하중 벡터이다. 지진 해석의 경우 지반 운동 $\ddot{u}_g(t)$ 은 다음과 같은 외력 벡터로 표현 가능하다.

$$\mathbf{f}(t) = -\mathbf{M}\mathbf{r}\ddot{u}_g(t) \quad (2)$$

여기에서 $\mathbf{r}$ 은 지반운동의 방향성을 나타내는 영향 벡터이다. 주파수 영역 해석은 식 (1)에 푸리에 변환을 적용하여 다음과 같이 주파수 영역에서 해를 구하게 된다.

$$\mathbf{S}(\omega)\hat{\mathbf{u}}(\omega) = \hat{\mathbf{f}}(\omega) \quad (3)$$

여기에서 $\mathbf{S}(\omega) = -\omega^2\mathbf{M} + i\omega\mathbf{C} + \mathbf{K}$ 이며, 동적강성행렬이다. $\hat{\mathbf{u}}(\omega)$ 와 $\hat{\mathbf{f}}(\omega)$ 는 각각 $\mathbf{u}(t)$ 와 $\mathbf{f}(t)$ 를 푸리에 변환한 주파수 영역에서의 변위 벡터와 하중 벡터이다. 일반적으로 지진 해석에 대한 주파수 영역해석은 다수의 진동수 성분에 대해 식 (3)의 해를 구해야 한다. 본 연구에서는 El Centro 지진을 가정하였으며 1Hz 간격으로 25Hz까지 식 (3)을 구성하여 해를 구하였다. 실제 지진해석에서는 보다 짧은 진동수 간격으로 해석할

필요가 있으나, 본 연구에서는 단순히 제안한 솔버의 성능 비교를 위해 1Hz 간격으로 선택하였다. 동적강성행렬 $\mathbf{S}(\omega)$ 는 대칭행렬이지만 indefinite 행렬이다. 따라서 CombinationS, CombinationD 솔버에 사용된 Krylov 방법은 CGS 알고리즘을 적용하였다. 감쇠행렬은 Rayleigh damping 1%를 적용하였으며, Table 2와 Table 3은 해석 모델의 고유진동수를 정리한 것이고, Fig. 8은 L-0.2와 S-0.2의 모드 형상을 도시한 것이다.

Intel Xeon(R) E5-2687W v4@3.00GHz CPU 2기와 256GB 메모리를 장착한 시스템에서 해석을 수행되었으며, 시스템의 전체 코어수는 24개이다. 직접 회소 솔버로는 Intel MKL 라이브러리에서 제공하는 PARDISO를 사용하였으며, Intel MKL 라이브러리의 회소행렬루틴으로 행렬-벡터 곱셈 연산 등을 구현하여 병렬화하였다. CGS 알고리즘의 수렴성은 $\|\mathbf{r}_j\|/\|\mathbf{b}\| < \epsilon$ 를 부과하였으며 수렴 허용치 ϵ 으로 10^{-5} 를 적용하였다. CombinationS 알고리즘에서 최대반복회수는 PARDISO에서 제안한 150을 적용하였다.

Table 4와 Table 5는 24 코어 및 8 코어가 사용했을 때 주파수 영역 해석에서 솔버에서 소요된 시간을 나타낸 것이고, Fig. 9와 Fig. 10은 CombinationS, CombinationD를 직접 솔버 대비 소요된 시간으로 정규화하여 나타낸 것이다. CombinationD는 해석 모델이 해석 모델이 커질수록, CPU 코어를 작게 사용할수록 직접 솔버와 비교할 때 속도 증가 효과가 큰 것을 알 수 있다. 또한 half band-width가 큰 S-x 모델에서 속도 증가 효과가 크다. 이와 같은 속도 증가 효과는 직접 솔버의 행렬 분해 단계에 소요되는 시간과 Krylov 반복 단계를 1회 수행하는데 소요되는 시간이 해석 모델이 클수록, CPU 코어가 작을수록 half band-width가 클수록 크게 차이 나기 때문이다. 반면에 CombinationS는 모든 경우에서 오히려 직접 솔버보다 더 많은 시간이 소요된다. CombinationS의

Table 2 Result of natural frequency analysis for L-x model

	Natural Frequency				Remark
	L-2	L-1	L-0.4	L-0.2	
1 st	1.02296	1.02319	1.02369	1.02432	first flexural mode for weak axis
2 nd	2.03262	2.03275	2.03303	2.03339	first flexural mode for strong axis
3 rd	6.34134	6.34283	6.34603	6.3501	second flexural mode for weak axis
4 th	12.22	12.2209	12.2229	12.2253	second flexural mode for strong axis

Table 3 Result of natural frequency analysis for S-x model

	Natural Frequency				Remark
	S-2	S-1	S-0.4	S-0.2	
1 st	4.0759	4.07659	4.07786	4.07965	first flexural mode for weak axis
2 nd	7.9679	7.96836	7.96919	7.97032	first flexural mode for strong axis
3 rd	24.4988	24.5035	24.5122	24.5244	second flexural mode for weak axis
4 th	30.8826	30.887	30.8951	30.9066	first twisting mode

Fig. 8 Mode shapes

효율이 나쁜 것은 고유진동수 근처에서 시스템 행렬 S 가 널 (null) 행렬에 접근하므로 행렬의 조건(condition)이 나빠져 150회의 Krylov 반복 계산에 해가 수렴되지 않고, 150회의 반복을 수행하는 시간은 직접 솔버로 행렬 분해하는 시간보다

Table 4 Wall time of frequency domain analysis - 24 cores used(unit : seconds)

Model	Direct Solver	CombinationS	CombinationD
L-2	5404.19	14006.8	4048.54
S-2	9683.81	10063.4	4547.21
L-1	1472.26	3837.26	1339.25
S-1	2372.59	3211.89	1409.62
L-0.4	267.711	1246.32	419.974
S-0.4	486.234	949.626	328.313
L-0.2	87.8953	633.893	124.805
S-0.2	126.402	473.368	136.353

Table 5 Wall time of frequency domain analysis - 8 cores used (unit : seconds)

Model	Direct Solver	CombinationS	CombinationD
L-2	7773.28	14143.1	4658.56
S-2	15107.6	9915.02	5170.11
L-1	2204.94	5062.54	1571
S-1	3704.24	3228.46	1617.67
L-0.4	476.937	1787.11	421.134
S-0.4	771.202	995.249	390.13
L-0.2	157.799	642.895	170.255
S-0.2	236.892	771.947	140.812

오히려 크기 때문이다. CombinationD는 런타임에서 최대 반복 회수를 결정하기 때문에 수렴성이 나쁜 경우, 적은 회수의 Krylov 반복 회수를 수행한 후 직접 솔버로 전환하게 된다. 결과적으로 PARDISO에서 제안하는 CombinationS 방법은 수렴성에 문제가 있는 시스템에 적용할 경우 효율이 나쁨을 알 수 있으며, CombinationD가 큰 해석모델에서 계산 효율이 뛰어나다.

Fig. 10과 Fig. 11은 모델의 크기 작은 L-0.2, S-0.2에 대해 계산에 참여하는 코어수를 달리하여 솔버에서 소요되는 시간을 비교한 것이다. 수렴성에 문제가 있는 CombinationS

Fig. 9 Speedup over direct solver in frequency domain analysis - 24 cores

Fig. 10 Speedup over direct solver in frequency domain analysis - 8 cores

Fig. 11 Wall time of solvers according to number of cores for model L-0.2 in frequency domain analysis

Fig. 12 Wall time of solvers according to number of cores for model S-0.2 in frequency domain analysis

는 배제하였다. 그림에서 8~12 코어까지는 CombinationD의 속도 증가 효과가 있으며 이후 직접 솔버의 소요시간에 접근하는 것을 알 수 있다. 이와 같은 결과는 직접 솔버에서 행렬 분해 단계는 코어 수에 따라 병렬화가 잘되어 있는 반면, 솔루션 단계는 상대적으로 병렬화가 잘되지 않는데(Cho, *et al.*,

2017), Fig. 2에 나타난 것과 같이 Krylov 반복에 소요되는 시간은 솔루션 단계를 수행하는데 크게 의존하기 때문이다.

4. 결 론

이 논문에서는 미세하게 변화하는 시스템 행렬을 대상으로 연속적으로 해를 구해야 하는 문제인 주파수 영역 해석에 효율적으로 적용 가능한 반복-직접 희소 솔버 조합 기법을 제안하였다. 제안된 기법은 실행시간에 최대 Krylov 반복 회수를 계산하여 최대 반복 Krylov 회수를 설정하는 간단한 알고리즘이다. 제안된 방법의 효과를 검증하기 위해 여러 규모의 유한요소 모델에 대해 주파수 영역 해석과 비선형 해석 문제를 구성하고, 순수하게 직접 희소 솔버를 적용한 경우와 반복-직접 희소 솔버 조합 기법을 적용한 경우에 대한 해석 시간을 비교하였다. 수치 실험 결과 대규모 유한요소 모델의 주파수 영역 해석에서 제안된 간단한 알고리즘의 도입으로 계산시간이 상당히 단축되는 것을 확인할 수 있었다.

감사의 글

본 연구는 산업통상자원부(MOTIE)와 한국에너지기술평가원(KETEP)의 지원을 받아 수행한 연구 과제입니다(No. 20161520101130).

References

- Bai, Z., Demmel, J., Dongarra, J., Ruhe, A., van der Vorst, H. (Eds.) (2000) *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, Society for Industrial and Applied Mathematics.
- Bathe, K.J. (1995) *Finite Element Procedures*, Prentice Hall.
- Cho, J.-R., Cho, K. (2017) Design Considerations on Large-scale Parallel Finite Element Code in Shared Memory Architecture with Multi-Core CPU, *J. Comput. Struct. Eng. Inst. Korea*, 30(2), pp.127~135.
- Davis, T.A. (2006) *Direct Methods for Sparse Linear Systems*, Siam.
- Intel Math Kernel Library <http://software.intel.com/en-us/intel-mkl>.
- LIS <https://www.ssisc.org/lis/>
- PARDISO <http://www.pardiso-project.org>
- Saad, Y. (2003) *Iterative Methods for Sparse Linear Systems*, Siam.

요 지

대규모 유한요소 모델을 빠르게 해석하기는 위해서 병렬 회소 솔버를 필수적으로 적용해야 한다. 이 논문에서는 미세하게 변화하는 시스템 행렬을 대상으로 연속적으로 해를 구해야 하는 문제에서 효율적으로 적용가능한 반복-직접 회소 솔버 조합 기법을 소개한다. 반복-직접 회소 솔버 조합 기법은 병렬 회소 솔버 패키지인 PARDISO에 제안 및 구현된 기법으로 새롭게 행렬값이 갱신된 선형 시스템의 해를 구할 때 이전 선형 시스템에 적용된 직접 회소 솔버의 행렬 분해(factorization) 결과를 Krylov 반복 회소 솔버의 preconditioner로 활용하는 방법을 의미한다. PARDISO에서는 미리 설정된 반복 회수까지 해가 수렴하지 않으면 직접 회소 솔버로 해를 구하며, 이후 이어지는 갱신된 선형 시스템의 해를 구할 때는 최종적으로 사용된 직접 회소 솔버의 행렬 분해 결과를 preconditioner로 사용한다. 이 연구에서는 첫 번째 Krylov 반복 단계에서 소요되는 시간을 동적으로 계산하여 최대 반복 회수를 설정하는 기법을 제안하였으며, 주파수 영역 해석에 적용하여 그 효과를 검증하였다.

핵심용어 : 대규모 유한요소 모델, 주파수 영역 해석, 회소 솔버, 반복-직접 회소 솔버 조합 기법