Modelowanie matematyczne w nauce i technice

LAB 09. Iteracyjne metody rozwiązywania układów równań liniowych

Wstęp

Istnieją dwa podstawowe sposoby rozwiązywania układów równań liniowych postaci **Ax=b** – metody bezpośrednie, które badane były podczas poprzednich zajęć, oraz metody iteracyjne. W metodach bezpośrednich, będących w praktyce najczęściej wariantami eliminacji Gaussa, po wykonaniu pewnej liczby operacji otrzymuje się rozwiązanie o dużej dokładności (błędy pojawiają się tylko na skutek skończonej dokładności obliczeń komputerowych i propagacji błędów zaokrągleń – dlatego obliczenia zawsze są przeprowadzane na liczbach podwójnej precyzji, co pozwala uzyskać wyniki z dokładnością co najmniej kilku cyfr znaczących).

Metody iteracyjne uzyskują rozwiązanie w szeregu kroków obliczeniowych (iteracji), zaczynając od pewnego wektora startowego $\mathbf{x_0}$ (*initial guess*) i uzyskując po każdym z kroków kolejne przybliżenia rozwiązania. Miarą zbieżności dla kolejnych rozwiązań $\mathbf{x_k}$ jest norma z wektora residuum układu równań, obliczanego dla $\mathbf{x_k}$ jako: $\mathbf{r_k = b - Ax_k}$. Zakłada się, że kiedy residuum układu równań dla kolejnych iteracji zmierza do 0, to także błąd rozwiązania zmierza do zera (jeśli rozwiązanie dokładne oznaczymy jako $\mathbf{x^*}$, a błąd rozwiązania $\mathbf{x_k}$ jako $\mathbf{e_k=x^* - x_k}$, to dzięki temu, że $\mathbf{Ax^*=b}$, otrzymuje się liniową zależność wiążącą błąd z residuum: $\mathbf{r_k = b - Ax_k = Ax^* - Ax_k = A(x^* - x_k) = Ae_k}$).

Znając normę residuum, nie znamy normy błędu, jednak z liniowej zależności między oboma wektorami możemy wnioskować, że kiedy norma residuum maleje wielokrotnie, to także norma błędu maleje w podobny sposób. Stąd częstym kryterium zatrzymania iteracji jest albo osiągnięcie przez normę residuum $||\mathbf{r_k}||$ określonej małej wartości $\mathbf{||r_k|| < e_{ABS}}$ (absolutna redukcja residuum), albo redukcja o określony czynnik początkowego residuum $\mathbf{r_0 = Ax_0 - b}$, $\mathbf{||r_k||/||r_0|| < e_{REL}}$ (względna redukcja residuum).

Najpopularniejszymi obecnie metodami iteracyjnego rozwiązywania układów równań liniowych są tzw. metody podprzestrzeni Kryłowa (rozwiązujące pewien problem optymalizacyjny), których typowymi przykładami są metody sprzężonych gradientów i GMRES. Metody podprzestrzeni Kryłowa zbiegają się tym lepiej im macierz układu jest bliższa macierzy jednostkowej (mającej optymalne uwarunkowanie), stad często w algorytmie metody pojawiają się dodatkowe operacje tzw. poprawa uwarunkowania macierzy (*preconditioning*).

Jedną z najważniejszych cech decydujących o wydajności i praktycznej użyteczności każdego z iteracyjnych solwerów liniowych (solwer – program rozwiązujący zadany problem), jest szybkość zbieżności – determinująca w ilu iteracjach solwer osiągnie wymaganą redukcję residuum układu. Ostateczna wydajność zależy także od czasu realizacji pojedynczej iteracji oraz od czasu przygotowania struktur danych do wykonywania iteracji (ten ostatni czas może być istotny dla wszelkich bardziej złożonych metod, np. z wyrafinowanymi *preconditionerami*, takimi jak algorytmy *multigrid* albo algorytmy niekompletnego rozkładu LU (*incomplete LU factorization*),

Do wyrażenia szybkości zbieżności można użyć liczby iteracji, **`N_it`**, prowadzących do osiągnięcia zbieżności (zakładanej redukcji normy residuum), można także obliczyć uśredniony po wszystkich iteracjach stosunek norm kolejnych residuów układu (nazywany dalej uśrednioną redukcją norm). Jeśli szybkość zbieżności w pojedynczej (k-tej) iteracji wyrażona zostanie wzorem: $\mathbf{conv\_rate\_at\_k\text{-}th\_iteration = ||r_{k+1}||/||r_k||}$, to ostateczny stosunek residuum po *k*-iteracjach do residuum początkowego $\mathbf{||r_k|| / ||r_0||}$ będzie można obliczyć jako iloczyn szybkości zbieżności w kolejnych iteracjach. W drugą stronę można obliczyć uśrednioną szybkość zbieżności jako: $\mathbf{average\_conv\_rate = (||r_k|| / ||r_0||)^{(1/Nit)}}$ (co prowadzi do wzoru: $\mathbf{||r_k|| / ||r_0|| = average\_conv\_rate^{N\_it}}$, wiążącego szybkość zbieżności z liczbą iteracji **`N_it`**). Im współczynnik **average_conv_rate** bliższy 0, tym zbieżność szybsza (wystarcza mniej iteracji do osiągnięcia założonej redukcji residuum, im bliższy 1, tym zbieżność wolniejsza; dla wartości **average_conv_rate** > 1 iteracje nie zbiegają się, solwer jest rozbieżny dla danego układu równań.

W ramach zajęć będziecie Państwo badać różne metody iteracyjne przedstawione na wykładzie: Jacobiego, Gaussa-Seidla orazGMRES – jako przykład metody podprzestrzeni Kryłowa, z różnymi

wariantami poprawy uwarunkowania macierzy układu równań. Testowane będą szybkość zbieżności normy residuum i czas uzyskiwania rozwiązania. Otrzymane wyniki posłużą do porównania z algorytmem PARDISO bezpośredniego rozwiązywania układu równań, badanym na poprzednich zajęciach.

1. **Zadanie 1. (obowiązkowe) Uruchomienie zadania z iteracyjnym solwerem układów równań liniowych.**

   1.1. Proszę utworzyć katalog ***lab_09***

   1.2. Wszystkie obliczenia w ramach laboratorium należy wykonywać na tych samych siatkach, który służyły do testowania solwera bezpośredniego podczas poprzedniego laboratorium (siatka początkowa mniej więcej pomiędzy 200 a 3**0**0 węzłów (wierzchołków), siatka po trzech adaptacjach jednorodnych z liczbą wierzchołków w zakresie 50 000 – 100 000)

   - Jako wstęp do ćwiczeń należy skopiować do katalogu ***lab_09*** pliki sterujące (problem_heat.dat, bc_heat.dat, plik siatki) z przykładu z poprzedniego laboratorium, pamiętając o nazwie (np. „test_solwerów”), która gwarantuje, że nie będą **włączały się warianty zadawania źródeł ciepła i inne dokonane indywidualnie wcześniej modyfikacje kodu**
     - w przeciwieństwie do solwera bezpośredniego zachowanie solwera iteracyjnego może istotnie zależeć od rozwiązywanego zadania (np. czy istnieje silne skondensowane źródło ciepła, czy występują nieliniowości itp.)
     - dlatego w początkowej fazie laboratorium **konieczne** jest badanie solwera w przypadku zadania standardowego (tak jak w lab 06 – **bez źródeł ciepła, nieliniowych materiałów i warunków brzegowych itp**.

   1.3. Następnie należy uruchomić program w istniejącej konfiguracji z solwerem bezpośrednim, rozwiązać zadanie i zanotować wyniki otrzymane przez solwer

   - wystarczające jest zanotowanie konkretnych wartości wybranych stopni swobody obliczonych przez solwer bezpośredni, **nie będących wartościami zadanych warunkow Dirichleta,** drukowanych jako np.:

     ```
     Solution in struct 0, block 3, dof_ent 7, posglob=3
           308.0015926661
     ```

   - można także sprawdzić jakościową poprawność rozwiązania wizualizując je w programie ParaView

   1.4. Pierwszym krokiem wykorzystania solwerów iteracyjnych jest skopiowanie ze strony przedmiotu do katalogu roboczego z plikami konfiguracyjnymi przykładowego problemu, pliku konfiguracyjnego dla interfejsu programu ModFEM z solwerami iteracyjnymi, **`mkb.dat`**.

   1.5. W pliku problemowym **`problem_heat.dat`** należy zaznaczyć opcję wyboru wbudowanego solwera iteracyjnego ModFEM (z podstawową metodą GMRES):

   ```
   linear_solver_type = 1; // int:
   ```

   a także nazwę pliku konfiguracyjnego solwera

   ```
   solver_file = "mkb.dat"; // solver_filename
   ```

   1.6. Po dokonaniu zmian należy uruchomić program, rozwiązać zadanie i sprawdzić poprawność wyniku (przez porównanie z wynikiem solwera bezpośredniego)

   a) najdokładniejszym sprawdzeniem jest porównanie konkretnych wartości wybranych stopni swobody obliczonych przez solwer i drukowanych jako (przykładowo):

   ```
   Solution in struct 0, block 3, dof_ent 7, posglob=3
         308.00016859953
   ```

b) należy zwrócić uwagę na wydruki (bezpośrednio przed wartościami wybranych obliczonych stopni swobody), przykładowo:

```
Convergence in GMRES after 15 iterations!!!
Total convergence rate (average decrease in residual per iteration) 0.347699
Norm of preconditioned residual 0.000622893, relative decrease 0.00000013123
After solving a system of 330 linear equations - norm of residuum:
      L2 =    0.039030809206188, MAX =    0.019712119716132
```

Wydruki informują o istotnych z punktu widzenia wydajności parametrach, takich jak: liczba iteracji do osiągnięcia zbieżności (założonej redukcji residuum) i **uśredniona redukcja norm, określająca szybkość zbieżności (*Total convergence rate (average decrease in residual per iteration)*)**. Podają także dwa parametry dokładności: normę z residuum po zastosowaniu preconditionera oraz normę ze standardowego residuum . Szczególnie ta ostatnia (**MAX**) jest istotna.

Norma L2 oznacza wybór normy średnio-kwadratowej (standardowej dla wektorów – pierwiastek z sumy kwadratów współrzędnych, czyli wartości stopni swobody), norma maksymalna MAX, wartość bezwzględną maksymalnej różnicy między współrzędnymi wektora.

Norma MAX może być porównana z różnicami wartości stopni swobody obliczonych przez solwer bezpośredni (przyjętych jako wartości dokładne) i obliczonych przez solwer iteracyjny (dla podanych powyżej przykładowych wydruków różnica wartości wynosi: **308.00016859953 – 308.0015926661 = 0,001424067,** błąd względny ok. **0,001424067/308 = 4.e^{-6}**; przyjmując zamiast różnicy dla konkretnego stopnia swobody wartość normy MAX: **MAX = 0.019712119716132**, dostajemy błąd względny ok. **0.019712119716132/308 = 64.e^{-6}** . Błąd względny będzie zazwyczaj większy (czasem nawet o kilka rzędów wielkości, ze względu na relacje błędu i residuum) od założonej dokładności względnej solwera iteracyjnego, zadanej jako warunek zbieżności dla względnej redukcji residuum.

2. **Zadanie 2 (obowiązkowe – na 3.0). Badanie dokładności i wydajności obliczeń metodami iteracyjnymi**

2.1. Pozostawiając jako aktywny terminal, w którym uruchamiany jest program **MOD_FEM_heat_prism_std**, należy w odrębnym terminalu niezależnie rozpocząć analizę i edycję pliku **mkb.dat**

a) (niezależność można oczywiście uzyskać w dowolny sposób, który pozwoli na, wygodne z punktu widzenia realizacji zadań równoległe edytowanie pliku **mkb.dat** i uruchamianie obliczeń z przeglądaniem wyników)

b) w ćwiczeniu najlepiej jest wykorzystać sposób działania ModFEM, polegający na wczytywaniu od nowa zawartości pliku **mkb.dat** przy każdorazowym rozwiązywaniu zadania (zastosowaniu opcji **'s'** z menu głównego)

c) rozwiązanie tego samego zadania różnymi wariantami solwera ModFEM można uzyskać przez wielokrotne użycie opcji **'s'**, każdorazowo dla innej zawartości pliku **mkb.dat**

d) w pliku **mkb.dat** znajduje się szereg wyjaśnień i wskazówek do przeprowadzania ćwiczenia

2.2. Porównanie działania solwerów należy przeprowadzić, podobnie jak to było w przypadku badania algorytmu przenumerowania) dla siatki dwukrotnie zaadaptowanej (ok. 6000-13000 wierzchołków = węzłów = stopni swobody = niewiadomych = funkcji bazowych)

a) dla każdego uruchomienia należy wypełnić wiersz tabelki **(po przeczytaniu uwag poniżej tabelki!)**, zawierającej następujące pozycje (dane należy pobrać z wydruków omawianych powyżej):

metoda iteracyjna	zbieżność	osiągnięta norma MAX residuum	liczba iteracji	uśredniona redukcja norm	czas rozwiązania	błąd względny vs. PARDISO
Jacobi	TAK?/NIE?					

przedostatnią i ostatnią kolumnę należy wypełnić tylko w przypadku uzyskania zbieżności **(jeśli nie uda się uzyskać zbieżności w 100 iteracjach, a proces jest zbieżny – norma residuum maleje – można zwiększyć liczbę iteracji, w przypadku bardzo szybko realizowanych iteracji nawet ponad stukrotnie)**

[Porównanie z solwerem PARDISO należy przeprowadzić wybierając wartość stopnia swobody (**`Solution in struct ...`**) **nie będącą wartością zadanych warunkow Dirichleta ,** np. o największej różnicy z rozwiązaniem iteracyjnym i obliczając błąd względny:

`( rozwiązanie_PARDISO - rozwiązanie_GMRES ) / rozwiązanie_PARDISO`

(uwaga przed przystąpieniem do obliczeń różnicy należy uruchomić zadanie na podwójnie zaadaptowanej siatce z PARDISO i uzyskać wyniki porównawcze)

[uwaga: ze względu na korzystanie wielu osób z serwera wyniki dotyczące czasu obliczeń mogą być znacznie zaburzone, dla uzyskania bardziej wiarygodnych wyników, obliczenia powinny być powtórzone kilka razy (opcja **`'s'`**) i jako wartość do tabelki wpisany czas najkrótszy (zaburzenia zawsze zwiększają czas obliczeń)]

b) każde uruchomienie jest związane z inną wersją solwera iteracyjnego, wybieraną poprzez ustalenie parametrów oznaczanych w komentarzach w pliku **`mkb.dat`** jako: wybór_solwera i wybór_preconditionera (dodatkowo niektóre z wariantów wymagają konkretnego sposobu przechowywania macierzy układu, ustalanego przez dobór parametru opisywanego w **`mkb.dat`** jako wybór_formatu_przechowania

c) poprawność zadania parametrów można sprawdzić na wydruku produkowanym przez program po uruchomieniu procedury rozwiązania (opcja **`'s'`** w menu), kilka linii poniżej menu głównego (poniżej wydruk dla pliku **`mkb.dat`** ze strony):

```
Initiated data for solver (id = 0): 1 mesh levels with parameters:
Level: 0
Solver type 1, GMRES type  0, Krylbas 50, Pdeg -1
Preconditioner 4, Number of sweeps 1, Block type/size 1, Storage_type 1
Max_iter 1000, Conv_type 0, Conv_meas    0.000001000000000
Monitor 3, Nr_pre_smmoth 1, Nr_post_smooth 1, ILU_k 0
```

a) w wydruku powyższym istotne są parametry określające wybrany algorytm: **`Solver type, Preconditioner, Storage_type`**, a także parametry zadanej dokładności solwera: **`Max_iter`** (ograniczenie maksymalnej liczby iteracji), **`Conv_type`** (zawsze zero – względna redukcja residuum ukladu), **`Conv_meas`** (zadana dokładność – tolerancja błędu). Za pomocą parametru **`Monitor`** można sterować liczbą wydruków, natomiast pozostałe parametry służą szczegółowym wyborom w ramach różnych algorytmów, badanych w ramach laboratorium, a także bardziej zaawansowanych).

[w trakcie eksperymentowania można zwiększać maksymalną liczbę iteracji, **`Max_iter`**, nawet do 10000 – nie powinno to znacząco wydłużyć czasu rozwiązania, metody o wolnej zbieżności bardzo szybko realizują pojedyncze iteracje, metody wykonujące w pojedynczej iteracji znacznie więcej operacji, zbiegają się znacznie szybciej]

2.3. Jako pierwszą metodę proszę przetestować najprostszy algorytm Jacobiego.
Ustawienia w pliku **`mkb.dat`**:
wybór_solwera – 10
wybór_preconditionera – 1
(pozostałe parametry bez zmian – zwłaszcza wybór_formatu_przechowania 1)

2.4. Drugą testowaną metodą jest metoda Gaussa-Seidla:
wybór_solwera – 10
wybór_preconditionera – 2

(pozostałe parametry bez zmian – zwłaszcza wybór_formatu_przechowania 1)

2.5. W pozostałych przypadkach testowane będą warianty metody GMRES:

wybór_solwera – 1

z różnymi opcjami wyboru poprawy uwarunkowania macierzy:

a) brak poprawy uwarunkowania: wybór_preconditionera – 0 (wybór_formatu_przechowania -2)

b) poprawa poprzez zastosowanie algorytmu Jacobiego: wybór_preconditionera – 1 (wybór_formatu_przechowania -2)

c) poprawa poprzez zastosowanie algorytmu Gaussa-Seidla: wybór_preconditionera – 2 (wybór_formatu_przechowania -2)

d) poprawa poprzez zastosowanie algorytmu niekompletnego rozkładu LU: wybór_preconditionera – 4 (wybór_formatu_przechowania 1)

2.6. Po wypełnieniu tabeli proszę zanalizować zawarte w niej dane, wyciągając wnioski co do charakterystyk wydajnościowych różnych algorytmów (uwzględniając liczbę iteracji i czas rozwiązania układu, zależny także od czasu realizacji pojedynczej iteracji oraz od czasu przygotowania struktur danych do wykonywania iteracji)

[szczególnie istotne mogą być przypadki, kiedy mimo redukcji residuum o zadaną wartość, rozwiązanie solwerem iteracyjnym różni się istotnie od rozwiązania solwerem bezpośrednim - może to oznaczać bardzo źle uwarunkowany układ równań , z którym solwer iteracyjny ma problemy (bardzo duże residuum początkowe, którego nawet duże zmniejszenie nie oznacza uzyskania dokładnego rozwiązania - istotną wskazówką jest tu wielkość normy maksymalnej, która powinna być znacząco mniejsza od 1 :

After solving a system of ... linear equations - norm of residuum:

L2 = ..., MAX = ...

3. **Zadanie 3 (na 4.0). Porównanie charakterystyk solwera bezpośredniego PARDISO i wybranych solwerów iteracyjnych zaimplementowanych w ModFEM**

3.1. Dla solwera bezpośredniego z poprzednich ćwiczeń oraz dla dwóch najlepszych wariantów solwerów iteracyjnych (3.5 d oraz 3.5 c – są to solwery o teoretycznie najszybszej zbieżności i najlepszych czasach rozwiązania, choć dla konkretnych zadań mogą ustępować teoretycznie mniej optymalnym wariantom 3.5 a i b) należy stworzyć 2 tabelki, **zawierające dla czterech kolejnych siatek MES, uzyskanych przez kolejne jednorodne adaptacje siatki początkowej**, następujące parametry:

a) wykorzystanie pamięci: dla solwera bezpośredniego - liczba wyrazów niezerowych po dekompozycji (w obu czynnikach L i U), natomiast dla solwerów iteracyjnych - liczba wyrazów niezerowych w macierzy układu (może być wzięta z poprzedniego laboratorium lub z wydruku na początku procedury rozwiązywania - tylko dla formatu przechowywania -2 (CRS):

Allocated CRS matrix 0: nglob = 297, nloc=297 nnz = 4345, half bandwidth = 26

(dla solwera 3.5d (GMRES+ILU(0)) liczba wyrazów powinna być wzięta z wariantu 3.5c i dodatkowo pomnożona razy dwa ze względu na przechowywanie macierzy po dekompozycji ILU)

[przyjęte parametry są przybliżone – dla każdej metody można dodatkowo uwzględnić mniej istotne czynniki: oryginalną macierz dla solwera bezpośredniego, wektory niewiadomych i prawej strony dla każdego z solwerów itp.]

b) czas rozwiązania (wydruk **`Time total`** bezpośrednio przed menu głównym)
   - dla solwera 3.5c (GMRES+GS) może być wymagane zwiększenie liczby iteracji w celu uzyskania zbieżności, np. do 10000
   - **ze względu na obciążenie serwerów ModFEM jest uruchamiany w wersji sekwencyjnej (bez zrównoleglenia OpenMP) - dotyczy to także solwera iteracyjnego. Solver PARDISO (z biblioteki MKL) domyślnie działa równolegle - dla uczciwego porównania z solwerem iteracyjnym powinien zostać uruchomiony także sekwencyjnie. Można to osiągnąć przez ustawienie zmiennych środowiskowych sterujących liczbą wątków (należy to zrobić w terminalu, gdzie uruchamiany jest program, bezpośrednio przed rozpoczęciem obliczeń):**
     export OMP_NUM_THREADS=1
     export MKL_NUM_THREADS=1

solwer	siatka 1	siatka 2	siatka 3	siatka 4
PARDISO				
GMRES+GS				
GMRES+ILU(0)				

3.2. Na podstawie danych z tabelek należy stworzyć wykresy zależności badanych parametrów od liczby stopni swobody problemu (jak zwykle należy użyć skali logarytmicznej na obu osiach)

3.3. Proszę przeanalizować wykresy i odpowiedzieć na pytania:
a) jak szybko rosną wymagania pamięciowe różnych solwerów?
b) jak szybko rośnie czas wykonania dla różnych solwerów?
c) w którym momencie solwery iteracyjne prawdopodobnie uzyskają czas rozwiązania krótszy niż solwer bezpośredni?
   - aby odpowiedzieć na to pytanie, należy ekstrapolować krzywe na wykresach – najlepiej przedłużając linie pomiędzy wynikami dla siatek 3 i 4 (im większa siatka, tym mniejszy wpływ czynników zaburzających podstawowe tendencje złożoności obliczeniowej (np. takie jak rosnąca wydajność solwera bezpośredniego dla coraz większych zadań, która po osiągnięciu maksymalnej wydajności dla danej maszyny przestaje rosnąć)

4. Zadanie 4 (na 5.0). Badanie szybkości zbieżności solwerów iteracyjnych w zależności od rozmiaru zadania

4.1. Dla solwerów z poprzedniego zadania (GMRES+GS i GMRES+ILU(0)) należy pobrać szczegółowe dane o normie residuum (po poprawie uwarunkowania, *preconditioned residual*), w co 10-tej (GMRES+ILU(0)) i co 50-tej (GMRES+GS) iteracji podczas rozwiązywania układów równań na kolejnych siatkach MES.

a) **poza zwiększeniem dopuszczalnej liczby iteracji należy ustawić poziom_szczegółowości_wydruku na wartość dwa i liczbę wektorów Kryłowa na zadaną wartość: 10 dla GMRES+ILU(0) i 50 dla GMRES+GS.** Pozwoli to pobierać wartości residuum co 10 / 50 iteracji metody

4.2. Na podstawie uzyskanych danych, proszę stworzyć 2 wykresy (dla każdego z solwerów) pokazujące normę z residuum po poprawie uwarunkowania (*norm of preconditioned residual*) dla każdej (GMRES+ILU(0)) i dla co 50 (GMRES+GS) iteracji (w efekcie na każdym z wykresów 4 linie, po jednej dla każdej siatki)

a) jaką niekorzystną zależność pokazuje porównanie otrzymanych wykresów?
b) wyniki analizy wykresów dodatkowo wesprzyj porównaniem uśrednionej redukcji norm (*Total convergence rate (average decrease in residual per iteration)*) pokazującej szybkości zbieżności dla obu solwerów i kolejnych siatek
   - stwórz tabelkę jak w p. 3.1 (bez solwera PARDISO) i wypełnij wartościami parametru
c) który z badanych solwerów iteracyjnych wydaje się być najbardziej uniwersalny?

Podsumowanie realizacji zadań (poniższa tabelka ma znaleźć się w sprawozdaniu bezpośrednio po wnioskach, a przed załącznikami - numeracja punktów realizacji kolejnych kroków laboratorium i załączników ma odpowiadać numeracji poniższych zadań)

Zadanie (skrócony opis)	OCENA własna w % (0-100)	OCENA prowadzącego w % (0-100)
Zad. 1 Uruchomienie zadania z iteracyjnym solwerem układów równań liniowych.		
Zad. 2 Badanie dokładności i wydajności obliczeń metodami iteracyjnymi.		
Zad. 3 Porównanie charakterystyk solwera bezpośredniego PARDISO i wybranych solwerów iteracyjnych zaimplementowanych w ModFEM		
Zad. 4 Badanie szybkości zbieżności solwerów iteracyjnych w zależności od rozmiaru zadania		
ŁĄCZNIE (400):		
OCENA KOŃCOWA:	----------	

Sprawozdanie powinno zawierać opis realizacji wszystkich zadań zawartych w temacie, wraz z omówieniem podstaw teoretycznych, odpowiedziami na pytania, wydrukami kodu i plików konfiguracyjnych oraz zamieszczonymi zrzutami ekranu. Opis realizacji każdego zadania może kończyć się wnioskami wynikającymi z przebiegu realizacji, całe sprawozdanie powinno zawierać wnioski dotyczące całości tematu.