Я.М., Мукоед А.П. Решение нелинейных задач теории оболочек на ЭВМ. - Киев: Вища школа, 1983.- 286 с. 5. Пелех Б.Л. Обобщенная теория оболочек. - Львов: Вища школа, 1978. - 159 с. 6. Подстригач Я.С., Швец Р.Н. Термоупругость тонких оболочек. - Киев: Наукова думка, 1978.- 344 с. 7. Рикардс Р.Б. Метод конечных элементов в теории оболочек и пластин. - Рига: Зинатне, 1988.- 284 с.

Vahin P.P., Malets R.B., Shynkarenko G.A.

Quasilinearization of the thermoelasticity problem of elastic shells with deformable normal

In this paper we formulate the boundary-value problem of the static thermoelasticity of shear isotropic shells. The mathematical model is based on geometrically nonlinear shells theory with deformable normal. Physical relations have been adopted due to Dugamel-Neuman hypothesis. Strain-displacement relations acknowledged linear distribution of rotation tensor components respecting to the shell's thickness. The variational problem is solved by Finite Element Methods employing biquadratic isoparametric approximations of displacements.

Стаття надійшла до редколегії 14.12.1998

УДК 517.6

В.Д.Вовк

Проблеми застосування об'єктного підходу до програмної реалізації чисельних методів розв'язування початково-крайових задач

Потреби інженерної практики важко адаптуються до існуючого розмаїття розроблених наукою математичних моделей та методів їх дослідження. Конструктори воліють постійно працювати з однимдвома універсальними пакетами прикладних програм, аніж для кожної конкретної задачі шукати, вивчати і адаптувати спеціалізоване програмне забезпечення, не зважаючи на його високу ефективність. Застосування нових потужних чисельних методів, таких як, наприклад, метод скінчених елементів (МСЕ), в цілому дозволило уніфікувати програмне забезпечення стосовно геометрії досліджуваного об'єкта. Проте спроби підтримки в одному пакеті програм аналізу більшого числа математичних моделей та ще й кількома методами неминуче приво-

дять до критичного росту складності створення, використання та супроводу програмного забезпечення.

Відомо, що ефективним способом розробки складної системи (в тому числі і програмної) є застосування об'єктного підходу. Проте використання елементів об'єктно-орієнтованого програмування (ООП) в реалізаціях окремих алгоритмів (наприклад, чисельних схем) найчастіше не тільки не приводить до їх покращення, а часто супроводжується суттєвим зниженням показників ефективності програмного продукту. Причиною є малоефективність об'єктно-орієнтованого програмування окремих алгоритмів без об'єктного аналізу задачі в цілому. Останній передбачає першочергову розробку повної структури класів, що моделюють об'єкти предметної області задачі та взаємозв'язки між ними. Відомо небагато таких достатньо повних систем класів, що є загальноприйнятими і використовуються багатьма колективами розробників (наприклад MFC фірми Microsoft). Але разом з тим вони демонструють великі переваги єдиного підходу до розуміння основних класових типів, що фігурують у вихідній задачі.

Поняття методу нерозривно зв'язане з областю його застосування. Тому створення повної системи класів для реалізації конкретного чисельного методу, наприклад МСЕ, вимагає попереднього визначення кола математичних моделей для дослідження. В цілях прискорення роботи розробники найчастіше обмежуються однією моделлю, розраховуючи на швидке переналагодження програм під іншу модель за рахунок "силових" засобів ООП. Проте для опису поведінки складних реальних конструкцій, систем та процесів часто необхідне одночасне використання кількох математичних моделей (спряження масивного тіла з тонкою оболонкою, тонкі включення тощо). Отже доводиться "склеювати" в одне ціле кілька програмних комплексів. Нове програмне утворення здебільшого не має ні задовільної ефективності ні достатньої універсальності. Причиною цього є те, що повна система класів кожного конкретного чисельного методу є лише елементом більш загальної структури класів, оскільки всякий метод є похідним від більш загального, як, наприклад, МСЕ є частковим випадком методу Гальоркіна. Таким чином ця повна система класів є лише локально повною і повинна наслідуватись від більш глобальних класових структур. Вершиною ієрархії мав би стати клас з найбільш загальними абстрактними компонентами, що властиві всім без виключення чисельним методам. Зрозуміло, що побудова абсолютно повної системи класів чисельних методів розрахованої на довільну математичну модель є безмежно довгим процесом. Але для розробки системи класів конкретного методу важливо лише щоб він враховував особливості всіх вищих по ієрархії методів. Тому реальна необхідність є лише

у створенні верхівки загальної структури класів чисельних методів. Відсутність такої верхівки приводить до двох негативних наслідків:

- створення високоуніверсального програмного пакету можливе лише шляхом його нарощування програмними реалізаціями окремих математичних моделей, виконаними різними розробниками, проте ці реалізації практично неможливо звести в єдиний програмний продукт;
- розробка основних математичних типів і засобів інтерфейсу між користувачем і програмою виконується заново при створенні кожного нового програмного забезпечення, що приводить до невиправданого дублювання зусиль розробників і потреб в ресурсах комп'ютера.

Нижче запропоновано варіант верхівки повної структури класів чисельного аналізу з прикладом деталізації однієї гілки до рівня МСЕ.

Довільний процес в будь-якій досліджуваній системі описується трьома компонентами:

Дія (об'єкт) –> результат

Метою вивчення системи чи процесу є знаходження однієї з цих компонент при відомих двох інших. Математична модель процесу може бути записана у вигляді операторного рівняння:

$$A(u) = f, \qquad (1)$$

або еквівалентної йому задачі знаходження екстремальних точок функціоналу:

$$\Phi(u) \Rightarrow min\ (max) \qquad (2)$$

де A – деякий оператор, який відображає множину U можливих станів системи на множину F допустимих на неї навантажень, а Ф-функціонал, що визначає значення деякої характеристики системи. З огляду на евівалентність (1) та (2) нижче будемо розглядати лише варіант моделі (1).

У найбільш складних випадках оператор A є диференціальним або інтегральним, а шуканий стан u ≡u(x) системи – функцією набору $x=(x_1, x_2, ..., x_n)$ її основних параметрів, що належать деякій області Ω. Чисельне розв'язування (1) полягає у застосуванні деякої схеми дискретизації оператора A і області Ω, яка дозволяє звести задачу (1) до системи алгебраїчних рівнянь. У частковому випадку $u(x) \equiv x$, коли стан досліджуваної системи визначається просто набором параметрів x, задача (1) безпосередньо формулюється у вигляді системи алгебраїчних рівнянь. Тому далі цей тривіальний випадок розглядатися не буде.

Навантаження f на систему поділяється на дві складові: внутрішнє $\varphi \in R(\Omega)$ та зовнішнє $g \in G(\partial\Omega)$, тобто $f = \{\varphi, g\}$, а $R \bigcup G = F$. Перший тип навантаження діє на кожний елемент системи, другий – тільки на елементи, що лежать на границі Ω. Відповідно оператор A є об'єднанням двох операторів $\{L, B\}$ з областями визначення $H(\Omega) \bigcup S(\partial\Omega) = U$, що моделюють реакцію системи на внутрішнє і зовнішнє навантаження. Враховуючи сказане, модель (1) подамо у вигляді наступної крайової задачі :

$$\begin{cases} \textit{Задано} & L : H(\Omega) \Rightarrow R(\Omega), \quad B : S(\partial\Omega) \Rightarrow G(\partial\Omega), \\ & \varphi \in R(\Omega), \quad g \in G(\partial\Omega) \\ \textit{Знайти} & u \in U \;\mid\; Lu = \varphi, \quad Bu\big|_{\partial\Omega} = g \end{cases} \tag{3}$$

Поіменуємо позначення: L – оператор математичної моделі, B – оператор крайових умов, H,S – банахові простори з шуканою функцією u , R,G – простори функціоналів, що визначені на H і S відповідно, Ω – досліджувана область, яка у всіх практичних задачах належить суперпозиції евклідового простору R^n і часового інтервалу [0,T].

За побудовою математичної моделі слідує вибір і реалізація методу її дослідження. Далі отриманий розв'язок аналізується з допомогою різноманітних форм його представлення. Сказане приводить до висновку, що у вершині структури класів чисельних методів на одному рівні ієрархії знаходяться три базові класи :

Кожен з них лежить в основі розгалуженого дерева похідних класів. В позначеннях мови C++ нижче покажемо елементи їх реалізації і структуру найближчих рівнів ієрархії.

Використовуючи (3) маємо:

```
Class TMathModel
 {TOperator *L, *B;        // Рівняння моделі і крайових умов
  TFunctional * f, *g;     // Праві частини рівнянь і КУ
  TGeometry * Г;           // Досліджувана область
  ...
 }
```

Тут тип TOperator є вершиною наступної піраміди класів :

TFunctional інкапсулює узагальнене поняття функціоналу.

TGeometry лежить у вершині системи класів, що описують геометрію досліджуваних об'єктів.

Всюди нижче ми не приводитимемо повний набір методів (зокрема конструкторів) класів, щоб не загромоджувати викладки.

Базовий клас представлення розв'зків :

```
Class TSolution
 {double * result;          // Масив значень розв'язків
  public :
 virtual void show( ) = 0;  // Процедура зображення розв'язку
 };
```

Для даного класу в силу великої різноманітності методів і засобів відображення результатів розв'язування задач ієрархія похідних класів достатньо не обґрунтована і знаходиться на стадії вивчення.

Більш детально проаналізуємо клас, що узагальнює поняття методу:

```
Class TMethod
 {TMathModel model;         // Математична модель
  TSolution u;              // Шуканий розв'зок
  public :
  virtual void solve( ) = 0; // Процедура пошуку розв'язку
  ... }
```

Даний клас поділяється на три основні гілки :

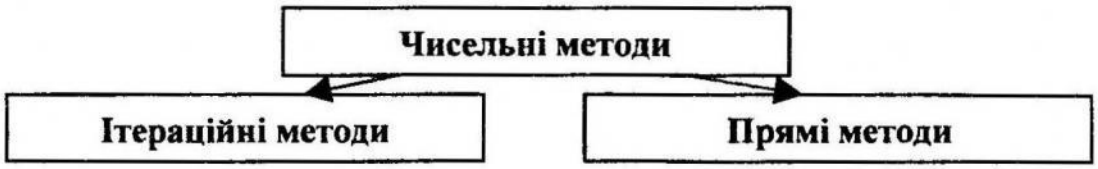
Признаком застосування чисельного методу є дискретизація розглядуваної області $\Omega \supset \Omega_h$, де h – параметр дискретизації, і на її основі заміна нескінченномірного простору U деяким скінченомірним його аналогом $U_n \subset U$ з базисом $\{\varphi_1,...,\varphi_n\}$. Отже клас чисельних методів має наступний вигляд:

```
Class      TNumericalMethod : public TMethod
  { int nelem;              // к-ть елементів дискретизації Ω_h
    Telem* elems;           // масив елементів дискретизації Ω_h
    TApproximation * a;     // простір апроксимацій U_n
   ... }
```

Чисельні методи полягають в заміні пошуку точного розв'язку вихідної задачі знаходженням параметрів α_i наперед вибраної апроксимуючої функції $\varphi(x,\alpha_1,...,\alpha_n)$, з умови задовільної її близькості до точного розв'зку. Тому базовий клас апроксимацій має наступний вигляд :

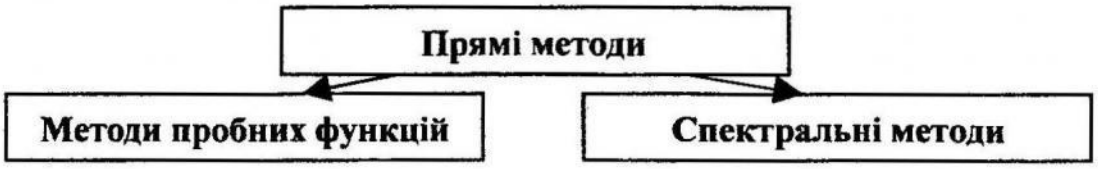
```
Class TApproximation
 {int n;                              // Кількість параметрів
  double * alpha;                     // Масив параметрів
  public :
  double value (double, ...) = 0;     // Значення апроксимації
  double valued (int,double, ...) = 0; // Значення похідних
  ... }
```

За способом пошуку параметрів α_i чисельні методи поділяються на дві групи :



де група ітераційних методів визначається заданням початкового наближення α_i з подальшим уточненням $\varphi_{j+1} = B(A)\varphi_j$. Конкретний ітераційний метод визначається вибором оператора B , який певним чином пов'язаний з оператором A вихідної задачі.

Прямі методи поділяються на групи вибором умови близькості апроксимуючої функції до точного розв'язку :



апроксимуюча функція $\varphi(x,\alpha_1,\ldots,\alpha_n)$ задовільняє оператор крайової задачі, а підбором параметрів α_i намагаються задовільнити крайові умови (маємо широкий набір спектральних методів);

апроксимуюча функція задовільняє крайовим умовам задачі, а підбором параметрів α_i намагаються задовільнити рівняння моделі (методи пробних функцій).

Методи пробних функцій характеризуються апроксимаціями виду $u^* = \sum_i \alpha_i \varphi_i$, тому :

```
Class TTestApproximation : public TApproximation
 {double (fun **) (double, ... );        // Набір пробних функцій
 }
```

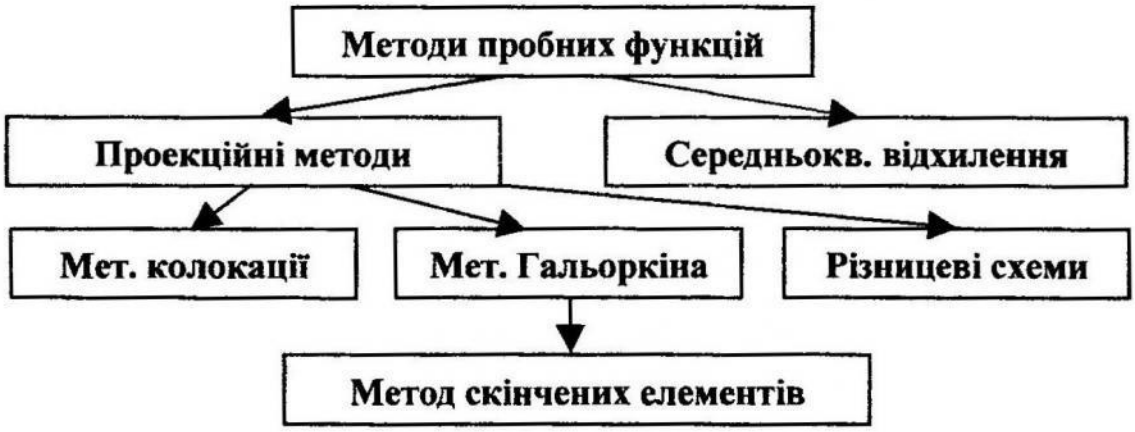
де функція value (double, ...) вже конкретизується.

На основі даних апроксимацій базуються дві групи методів, що відрізняються способом задовільняння рівнянь моделі :

$\int (F u^* - f)^2 \, d\Omega \Rightarrow min$ – середньоквадратичні методи

$\int (F u^* - f) \Psi_i \, d\Omega = 0$ – проекційні методи (зважених залишків)

де ψ_i – деякі вагові функції. Таким чином подальша ієрархія класів може бути представлена у вигляді :



Враховуючи сказане можемо записати :

```
Class TTestMethod : public TNumericalMethod
 {TCondition * c;   // Правила задовільняння оператору моделі
 ... }
```

Отже клас проекційних методів Class TProjectMethod : public TTestMethod відрізняється від батьківського лише конкретизацією :

```
Class TProjectCondition : public TCondition
{double (weight **) (double, ... );     // Набір вагових функцій
}
```

Підстановка weight = fun з класу TTestApproximation приводить до класу TGaliorkinMethod : public TProjectMethod

Наведена ієрархія класів в межах статті не претендує ні на достатню повноту структури ні на детальний опис реалізації. За мету даної роботи поставлено лише показ необхідності, можливості і основні напрямки розробки такої ієрархії.

Література.

1. Буч Г. Обьектно-ориентированное проектирование с примерами применения: Пер. с англ. – М.: Конкорд, 1992.- 519 с. 2. В.Д.Вовк, Р.Б.Петришин, Г.А.Шинкаренко. Верхівка системи класів програмної реалізації чисельних методів // Вісник Львів. ун-ту, сер. мех.-мат. - 1998.- Вип.50.- С. 48-51.

V.D.Vovk

The problems of object approach application to program implementation of the numerical methods of initial-boundary value problems solution.

Some problems of Object-Oriented Approach application to the development of numerical methods software have been investigated. It has been proved that its necessary to analyze the general structure of numerical methods class types for realization all the advantages of Object-Oriented Approach. A well-founded version of program classes top level structure is presented.

Стаття надійшла до редколегії 30.11.1998

УДК 533.6.013.42

В.М.Горлач, Я.В.Кондратюк

Чисельна модель акустичної взаємодії пружного тіла з рідиною. 4. Осесиметрична задача для ізотропних середовищ з в'язкістю

Дана праця присвячена чисельному моделюванню акустичної взаємодії осесиметричної системи в'язкопружне ізотропне тіло –