Санкт-Петербургский государственный университет
Кафедра информационных систем

Возженников Игорь Олегович

Магистерская диссертация

QR-алгоритм для лямбда-матриц в анализе устойчивости электронных схем

Направление 01.04.02
Прикладная математика и информатика
Магистерская программа: Математическое моделирование в задачах естествознания

Научный руководитель:
кандидат физ.-мат. наук,
доцент Еремин А. С.

Санкт-Петербург
2019

Содержание

Введение

В данной работе будет рассмотрен вопрос устойчивости линейных систем и, в частности, электронных схем. Устойчивость необходима для того, что понимать, прогнозировать и качественно оценивать решение системы.

Одним из наиболее важных и надежных критериев устойчивости системы является нахождение корней характеристического уравнения этой системы, то есть решение проблемы собственных значений.

В случае радиоэлектронной схемы, изначально она описывается нелинейной системой, и потому сначала проводится процесс линеаризации системы. После такой процедуры получается регулярный пучок матриц $D(\lambda) = \lambda B - A$, который уже можно проверить на устойчивость. И один из самых главных способов такого анализ — решение обобщенной собственной проблемой пучка матриц.

Обобщенная полная собственная проблема является одной из самых важных задач в анализе устойчивости радиоэлектронных схем.

Решение алгебраической проблемы собственных значений тесно связано с решением системы линейных обыкновенных дифференциальных уравнений с постоянными коэффициентами. Общая система однородных уравнений первого порядка с n неизвестными $y_1, y_2, \ldots, y_n$ может быть записана в виде:

$$B\frac{d}{dt}(y) = Cy, \tag{1}$$

где t — независимая переменная, y(t) — искомая вектор-функция решения; B и C — матрицы n-го порядка с постоянными коэффициентами. Если B — особенная, то левые части системы уравнений связаны линейным соотношением. Правые части должны удовлетворять тому же самому соотношению и, следовательно, y_i линейно зависимы. Систему уравнений, поэтому можно привести к системе низшего порядка.

Если B – неособенная, тогда уравнение (1) можно записать в виде (форма Коши, разрешенная относительно вектора производных):

$$\frac{d}{dt}(y) = Ay, \tag{2}$$

где

$$A = B^{-1}C$$

.

Предположим, что (2) имеет решение вида

$$y = xe^{\lambda t}, \tag{3}$$

где x — это вектор, не зависящий от t. Тогда

$$\lambda x e^{\lambda t} = A x e^{\lambda t},$$

и поэтому

$$\lambda x = Ax.$$

Следовательно, уравнение (3) дает решение уравнения (2), если X — собственное значение матрицы A, а x — соответствующий ему собственный вектор.

Если у A имеется r линейно независимых собственных векторов, то мы получим r линейно независимых решений (2), независимо от того, будут ли различны соответствующие собственные значения. Однако, уравнение (2) должно иметь n линейно независимых решений так, что если $r < n$, т. е. если некоторые элементарные делители A нелинейны, то (2) будет иметь решения не чисто экспоненциального типа.

Если же рассмотреть систему дифференциальных уравнений в области электронных схем, то ситуация усложняется нелинейностью системы в изначальном виде. И в этом случае матрица B всегда особенная.

В общем случае радиоэлектронная схема описывается нелинейной дифференциальной системой уравнений

$$B(x)\frac{d}{dt}x(t) = Ax(t) + g[x(t)] + f(t), x(t_0) = x_0, \tag{4}$$

где B(x) — матрица при векторе производных, элементы которой могут нелинейно зависеть от x порядка n в зависимости от выбора базиса пере-

менных; A — постоянная вещественная матрица коэффициентов порядка n; x(t) — вектор переменных, т.е. искомого решения, порядка n; g[x(f)] — нелинейная вектор-функция, зависящая от x(t); f(t) — вектор воздействий (сигналов); x_0 — вектор начальных условий в момент времени t_0.

Матрица B(x), как правило, вырождена. Системы уравнений, у которых вырождена матрица при производных, называют либо вырожденными системами, либо дифференциально-алгебраическими системами уравнений (ДАСУ).

В зависимости от способа формирования уравнений и применяемых математических моделей элементов в качестве переменных x(t) могут выступать узловые потенциалы, напряжения на элементах, токи, заряды и потокосцепления.

Разложим нелинейную вектор-функцию g[x(t)] в ряд Тейлора в окрестности точки x_0 и получим

$$g[x(t)] = g[x_0] + J(x_0) \times (x(t) - x_0) + R[x(t)],$$

где g(x_0) - постоянный вещественный вектор значений нелинейной вектор-функции в точке x_0 для времени t_0; J = J(x_0) - постоянная вещественная матрица коэффициентов (матрица Якоби), т.е. матрица первых частных производных вида [df_i/dx_j]: R[x(t)] – остаток ряда Тейлора для старших производных.

Предположим, что элементы матрицы B не зависят от x и она является матрицей с постоянными вещественными коэффициентами. В этом случае исходную нелинейную систему можно представить теперь в виде

$$B(x)\frac{d}{dt}x(t) = [A + J]x(t) + R[x(t)] + g(x_0) - Jx_0 + f(t).$$

Обозначим A = A + J и f*(t) = g(x_0) – Jx_0 + f(t) и рассмотрим нелинейную ДАСУ

$$B(x)\frac{d}{dt}x(t) = Ax(t) + R[x(t)] + f*(t), x(t_0) = x_0. \quad (5)$$

Если пренебречь нелинейными членами, т.е. вектором R[x(t)], то мы перейдем к линеаризованной системе уравнений, описывающей решение при таких малых f(t), что можно пренебречь влиянием нелинейностей (так называемый режим малого сигнала).

Как известно, для анализа устойчивости схемы в настоящее время существуют две группы критериев: алгебраические (критерий Ляпунова, Гаусса—Гурвица, Льенара—Шипара и др.) и частотные (критерии Найквиста, Михайлова и др.).

Из линеаризованной системы уравнений схемы (5) можно вычленить пару матриц B и A, в операторном виде связанных соотношением

$$D(p) = pB - A,$$

где p - параметр (оператор Лапласа).

В теории цепей и систем такую матрицу принято называть операторной матрицей системы, а в теории матриц — пучком матриц.

Определитель данной параметрической матрицы

$$det[pB - A] = a_m p^m + a_{m-1} p^{m-1} + ... + a_1 p + a_0 = a_m \prod_{i=1}^{m} (p - \lambda_i)$$

является полиномом от оператора Лапласа p и как всякий полином имеет корни λ_i. Принято этот полином называть характеристическим полиномом. Его корни в теории цепей называют собственными частотами схемы, а в теории матриц - собственными значениями пучка матриц.

Порядок характеристического полинома m не превышает размерности системы уравнений n, но если матрица B вырождена, то он обязательно будет меньше порядка системы уравнений.

Устойчивость систем уравнений (4) и (5) зависит от расположения корней характеристического полинома на комплексной плоскости. Сами корни могут быть либо вещественными, либо образуют комплексно-сопряженные пары.

Очевидно, собственные числа с положительной вещественной частью

будут говорить о неустойчивости системы и подробнее этот вопрос будет рассмотрен в 1 главе.

Таким образом, в этой работе поставлена задача решения обобщенной собственной проблемы (поиск чисел и векторов) и выбор для этого наилучшего метода.

Глава 1. Устойчивость линейных систем

1.1 Устойчивые компоненты решения неустойчивой системы

Линейная неоднородная система в общем виде

$$B\dot{x}(t) = Ax(t) + f^*(t).$$

И ее решение представляется в виде

$$x(t) = \sum_{k=1}^{m} [u_k v_k^T e^{\lambda_k t}] \cdot B \cdot x(0). \tag{6}$$

В общем случае устойчивость системы определяется набором собственных чисел пучка $(\lambda B - A)$ и совпадает с устойчивостью однородной системы.

Согласно критерию Ляпунова, система уравнений $B\dot{x}(t) = Ax(t)$ устойчива, если все корни лежат в левой комплексной полуплоскости, т.е. имеют отрицательные вещественные части. Если хотя бы один корень лежит в правой полуплоскости, т.е. имеет положительную вещественную часть, то система является неустойчивой. С точки зрения электронных схем в такой схеме возникает расходящийся во времени переходной процесс.

Да, в решение входит начальное условие и можно подобрать его таким, чтобы в него вообще не входили экспоненты с положительными показателями, однако. Такое решение будет неустойчивым по начальным данным, в том смысле, что все решения из его окрестности содержат эту неустойчивую компоненту и со временем уходят на бесконечность.

В итоге, поиск собственных чисел позволяет судить об устойчивости решения системы в первом приближении.

1.2 Устойчивые компоненты решения неустойчивой системы

Линейный анализ устойчивости лишь грубо позволяет оценить устойчивость нелинейной системы, даёт лишь первое приближение. Решения

нелинейных систем довольно часто ограничены, даже если в некоторые моменты времени (или даже почти всегда) линейное приближение оказывается неустойчивым. Так, например, линейная нестационарная система

$$\frac{dy_1(t)}{dt} = \sin(t) * y_2(t),$$
$$\frac{dy_2(t)}{dt} = \sin(t) * y_1(t)$$

в любой момент времени, кроме π*k имеет два равных по абсолютной величине, но противоположных по знаку собственных числа. Тем не менее её решение, например, при начальных данных $y_1(0) = 1$, $y_2(0) = 0$, очевидно ограничено для любого t

$$y_1(t) = \cosh(1) * \cosh(\cos(t)) - \sinh(1) * \sinh(\cos(t)),$$
$$y_2(t) = \cosh(\cos(t)) * \sinh(1) - \sinh(\cos(t)) * \cosh(1).$$

Однако даже в линейном анализе можно получить дополнительную информацию, которая позволит с большей осторожностью говорить о неустойчивости системы. Дело в том, что анализ устойчивости на основе вычисления собственных чисел пучка матриц позволяет определить, устойчива ли система в целом, то есть, устойчивы ли все её компоненты. В то же время набор собственных чисел сам по себе ничего не говорит об устойчивости отдельных компонентов системы. Рассмотрим простой пример.

$$\begin{gathered} \frac{dy_1(t)}{dt} = -y_1(t), \frac{dy_2(t)}{dt} = y_1(t) + y_2(t) \\ y_1(0) = y0_1, y_2(0) = y0_2, \\ y_1(t) = y0_1 \exp(-t) \\ y_2(t) = -\frac{y0_1 \exp(-t)}{2} + \exp(t)(\frac{y0_1}{2} + y0_2). \end{gathered} \tag{7}$$

Здесь неустойчивое слагаемое, содержащее $\exp(t)$, присутствует только во второй компоненте решения, тогда как компонента $y_1(t)$ устойчива при любых начальных данных. Очевидно, что в (7) решения можно найти последовательно: сначала решить первое уравнение (с единственным собственным числом -1), а затем, считая решение первого неоднородностью

для второго, решить второе уравнение с собственным числом 1, дающим неустойчивость.

В следующем примере такая структура неочевидна: матрица системы не имеет особенной формы, позволяющей выделить устойчивую подсистему. Её собственные числа -3, -2, -1 и 1.

$$\frac{dz_1(t)}{dt} = -7z_1(t) - 8z_2(t) + 4z_3(t) - 2z_4(t),$$
$$\frac{dz_2(t)}{dt} = 4z_1(t) + 5z_2(t) - 4z_3(t) + 2z_4(t),$$
$$\frac{dz_3(t)}{dt} = -2z_1(t) + z_2(t) - 5z_3(t) + 3z_4(t),$$
$$\frac{dz_4(t)}{dt} = -6z_1(t) - 3z_2(t) - 2z_3(t) + 2z_4(t)$$
$$z_1(0) = z0_1, z_2(0) = z0_2, z_3(0) = z0_3, z_4(0) = z0_4$$
$$z_1(t) = (-7z0_2 + 4z0_3 - 2z0_4 - 4z0_1)\exp(-t) + (z0_2 + z0_1)\exp(-3t) +$$
$$+(6z0_2 - 4z0_3 + 2z0_4 + 4z0_1)\exp(-2t),$$
$$z_2(t) = -(-7z0_2 + 4z0_3 - 2z0_4 - 4z0_1)\exp(-t) -$$
$$-(6z0_2 - 4z0_3 + 2z0_4 + 4z0_1)\exp(-2t),$$
$$z_3(t) = (z0_2 - z0_3 + z0_4)\exp(t) -$$
$$-\frac{(6z0_2 - 4z0_3 + 2z0_4 + 4z0_1)\exp(-2t)}{2} + 2(z0_2 + z0_1)\exp(-3t),$$
$$z_4(t) = 2(z0_2 - z0_3 + z0_4)\exp(t) + \frac{(6z0_2 - 4z0_3 + 2z0_4 + 4z0_1)\exp(-2t)}{2} +$$
$$+(-7z0_2 + 4z0_3 - 2z0_4 - 4z0_1)\exp(-t)$$

Обратим внимание, что y_1 и y_2 содержат только слагаемые с отрицательными собственными числами, а y_3 и y_4 — ещё и $\exp(t)$.

В ДАСУ подобная ситуация так же встречается, и предварительный анализ системы (подобный выделению устойчивой подсистемы в 7) тем более затруднён в силу наличия двух матриц.

Первый пример дифференциально-алгебраической системы:

$$2\frac{dy_2(t)}{dt} - \frac{dy_3(t)}{dt} - 5y_1(t) + 12y_2(t) - 6y_3(t) = 0,$$
$$\frac{dy_1(t)}{dt} - \frac{dy_2(t)}{dt} + \frac{dy_3(t)}{dt} + 4y_1(t) - 5y_2(t) + 2y_3(t) = 0,$$
$$2\frac{dy_1(t)}{dt} - 4\frac{dy_2(t)}{dt} + 3\frac{dy_3(t)}{dt} + 8y_1(t) - 12y_2(t) + 5y_3(t) = 0$$
$$y_1(0) = y0_1, y_2(0) = y0_2, y_3(0) = y0_3$$
$$y_1(t) = 2\exp(-t)y0_2 - \exp(-t)y0_3$$
$$y_2(t) = \exp(t)y0_1 + (2\exp(-t) - 3\exp(t))y0_2 + (-\exp(-t) + 2\exp(t))y0_3$$
$$y_3(t) = 2\exp(t)y0_1 + (2\exp(-t) - 6\exp(t))y0_2 + (-\exp(-t) + 4\exp(t))y0_3$$

Второй пример дифференциально-алгебраической системы:

$$2\frac{dy_1(t)}{dt} - 2\frac{dy_2(t)}{dt} + 2\frac{dy_3(t)}{dt} + \frac{dy_4(t)}{dt} + 13y_1(t) - 18y_2(t) + 20y_3(t) + 10y_4(t) = 0,$$
$$\frac{dy_1(t)}{dt} - \frac{dy_2(t)}{dt} + 2\frac{dy_3(t)}{dt} + \frac{dy_4(t)}{dt} + 3y_1(t) - 8y_2(t) + 8y_3(t) + 4y_4(t) = 0,$$
$$\frac{dy_1(t)}{dt} - 2\frac{dy_2(t)}{dt} + 3\frac{dy_3(t)}{dt} + \frac{dy_4(t)}{dt} + 3y_1(t) - 7y_2(t) + 7y_3(t) + 4y_4(t) = 0,$$
$$-2\frac{dy_1(t)}{dt} + 4\frac{dy_2(t)}{dt} - 4\frac{dy_3(t)}{dt} - \frac{dy_4(t)}{dt} - 18y_1(t) + 26y_2(t) - 28y_3(t) - 15y_4(t) = 0$$
$$y_1(0) = y0_1, y_2(0) = y0_2, y_3(0) = y0_3, y_4(0) = y0_4$$
$$y_1(t) = (6\exp(-2t) - 4\exp(-t))y0_1 + (-6\exp(-2t) + 4\exp(-t))y0_2 +$$
$$+(8\exp(-2t) - 6\exp(-t))y0_3 + (4\exp(-2t) - 3\exp(-t))y0_4,$$
$$y_2(t) = (-3\exp(-2t) + 4\exp(-t))y0_1 + (3\exp(-2t) - 4\exp(-t))y0_2 +$$
$$+(-4\exp(-2t) + 6\exp(-t))y0_3 + (-2\exp(-2t) + 3\exp(-t))y0_4,$$
$$y_3(t) = (-3\exp(-2t) + 4\exp(-t))y0_1 + (3\exp(-2t) - 4\exp(-t) - \exp(t))y0_2 +$$
$$+(-4\exp(-2t) + 6\exp(-t) + \exp(t))y0_3 + (-2\exp(-2t) + 3\exp(-t))y0_4,$$
$$y_4(t) = (-6\exp(-2t) + 4\exp(-t))y0_1 + (6\exp(-2t) - 4\exp(-t) + 2\exp(t))y0_2 +$$
$$+(-8\exp(-2t) + 6\exp(-t) - 2\exp(t))y0_3 + (-4\exp(-2t) + 3\exp(-t))y0_4$$

Приведённые примеры демонстрируют, что ряд компонент решения может быть устойчивым, даже в случае наличия в линейной системе соб-

ственных чисел с положительной действительной частью. В тех ситуациях, когда такими устойчивыми компонентами оказываются те напряжения и токи, которые в действительности интересуют пользователя, например, входное и выходное напряжения некоторой подсхемы, не следует категорически говорить о неустойчивости. Конечно, следует обратить внимание пользователя на наличие неустойчивых компонент, но указать на то, что эти неустойчивые компоненты являются внутренними токами или напряжениями некоторой подсхемы, и их неустойчивость в линейном приближении с высокой вероятностью не влияет на устойчивость схемы в целом (неустойчивость может быть следствием неточностей в моделях некоторых компонентов или вызвана линеаризацией). В таком случае ответ о поведении системы с учётом нелинейностей может дать только решение нелинейной ДАСУ во времени.

Таким образом, определение устойчивости отдельных компонентов решения линеаризованной системы может дать важную информацию для принятия инженером решения — изменять ли схему дальше или запускать трудоёмкий временной анализ.

Согласно формуле (6), чтобы определить, какие из собственных чисел входят в те или иные компоненты решения однородной системы, необходимо найти матрицы левых и правых собственных векторов. Соответственно, анализ матриц $[u_k v_k^T e^{\lambda_k t}] \cdot B$ для $\Re(\lambda_k) > 0$ позволит определить, какие компоненты устойчивы, а какие — нет.

В последующих главах мы рассмотрим методы нахождения собственных чисел, а также нахождения и построения собственных векторов.

Глава 2. Поиск собственных чисел как корней характеристического многочлена

2.1 Метод секущих

Метод секущих это интерполяционный алгоритм поиска корней полинома, который использует последовательность корней секущих линий для лучшей аппроксимации корня функции f. Метод секущих можно представить как конечно-разностное приближение метода Ньютона, хотя исторически они никак не были связаны.

Данные метод определяется следующим соотношением

$$x_n = x_{n-1} - f(x_{n-1})\frac{x_{n-1} - x_{n-2}}{f(x_{n-1}) - f(x_{n-2})} = \frac{x_{n-2}f(x_{n-1}) - x_{n-1}f(x_{n-2})}{f(x_{n-1}) - f(x_{n-2})}.$$

Как можно заметить из рекуррентного соотношения, метод секущих требует 2-е входящих значения x_0 и x_1, которые следует выбрать наиболее близко к корню.

Начиная с выходных значений x_0 и x_1, строится линия через точки $(x_0, \mathrm{f}(x_0))$ и $(x_1, \mathrm{f}(x_1))$. В виде «наклон-перехват» уравнение выглядит как

$$y = \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x - x_1) + f(x_1).$$

Корнем этого уравнения является $\mathrm{x} = x_2$ при $\mathrm{y} = 0$ $(\mathrm{f(x)} = 0)$:

$$x = x_1 - f(x_1)\frac{x_1 - x_0}{f(x_1) - f(x_0)}.$$

Так на каждой итерации находим очередное значение x_n до той степени, пока не получим желаемую точность, которая определяется разностью между x_n и x_{n-1}:

$$x_n = x_{n-1} - f(x_{n-1})\frac{x_{n-1} - x_{n-2}}{f(x_{n-1}) - f(x_{n-2})}.$$

2.2 Метод Мюллера

Данный метод является так же, как метод секущих интерполяционными алгоритмом для решения уравнений полинома, вида f(x) = 0. Кроме того, метода Мюллера основывается на методе секущих, но с тем отличием, что этот алгоритм использует 3 входные точки, вместо 2-ух (как у секущих).

Метод Мюллера строит параболу через 3 входные точки, после чего, точку пересечения параболы с осью X берет для следующего приближения.

Вначале берется 3 точки $(x_0,\ \mathrm{f}(x_0))$, $(x_{-1},\ \mathrm{f}(x_{-1}))$ и $(x_{-2},\ \mathrm{f}(x_{-2}))$ и вычисляется $(x_1,\ \mathrm{f}(x_1))$, затем по x_1, x_0 и x_{-1} получают $(x_2,\ \mathrm{f}(x_2))$ и т.д.

Решение находят построением параболы $y_n(\mathrm{x})$, проходящей через точки $(x_{n-1},\ \mathrm{f}(x_{n-1}))$, $(x_{n-2},\ \mathrm{f}(x_{n-2}))$ и $(x_{n-3},\ \mathrm{f}(x_{n-3}))$. Ее вид записывается в формуле Ньютона

$$y_n(x) = f(x_{n-1}) + (x - x_{n-1}) f[x_{n-1}, x_{n-2}] + (x - x_{n-1})(x - x_{n-2}) f(x_{n-1}, x_{n-2}, x_{n-3}),$$

где $\mathrm{f}[x_{n-1},\ x_{n-2}]$ и $\mathrm{f}[x_{n-1},\ x_{n-2},\ x_{n-3}]$ это разделенные разности. Далее формулу можно переписать, как

$$y_n(x) = f(x_{n-1}) + \omega(x - x_{n-1}) + f[x_{n-1}, x_{n-2}, x_{n-3}](x - x_{n-1})^2,$$

где

$$\omega = f[x_{n-1}, x_{n-2}] + f[x_{n-1}, x_{n-3}] - f[x_{n-2}, x_{n-3}].$$

В следующей итерации x_k дано как решение, ближайшее к x_{k-1} в квадратичном уравнении $y_k(\mathrm{x}) = 0$. И это дает общую формулу с помощью рекуррентного соотношения

$$x_n = x_{n-1} - \frac{2f(x_{n-1})}{\omega \pm \sqrt{\omega^2 - 4f(x_{n-1}) f[x_{n-1}, x_{n-2}, x_{n-3}]}}.$$

Стоит учитывать, что x_k может быть комплексным, даже если все

предыдущие значения были действительными. Это важное отличие от таких интерполяционных методов как, например, метод секущих и Ньютона, которые сохраняют получение действительных корней, если начальные условия заданы действительными числами.

2.3 Алгоритм вычисления характеристических корней

Этапы вычисления собственных значений пучка $(A-\lambda B)$ с помощью интерполяционных методов:

1. *Выяснить ранг матрицы* B;

Для начала узнаем количество собственных чисел в пучке (оно не равно размерности матриц). Ранг матрицы B равен количеству собственных чисел, найдем его, например, с помощью LU-разложения.

2. *Получение начального значения полинома*;

Значение полинома в ноле равно определителю матрицы A. Находим определитель, опять же, LU-разложением. Теперь переходим непосредственно к методам.

3. *Применение методов секущих и Мюллера для поиска корней.*

Во время поиска корней нужно совмещать методы секущих и Мюллера, так как полином пучка может иметь и действительные и комплексные корни. В таком случае итерация всегда начинается методом секущих, чтобы оставаться в поле действительных чисел, а переход к алгоритму Мюллера происходит после нескольких не сходящихся итерациях секущих (это, скорее всего, говорит о приближении к двум комплексно-сопряженным корням).

Такой подход является наиболее эффективным по экономии памяти, количеству итераций и времени получения корней полинома. Метод секущих требует меньше данных и вычислений, чем метод Мюллера, а также все вычисления проходят с действительными числами, без лишних трат на комплексную арифметику. Метод Мюллера помогает быстрее находить комплексно-сопряженные корни, потому что методу секущих потребуется намного больше итераций, чтобы сойтись к таким корням.

Глава 3. Степенные методы

3.1 Метод прямых итераций (степенной метод)

Для начала рассмотрим матрицы, имеющие линейные элементарные делители. Для любой такой матрицы A имеем

$$A = Xdiag(\lambda_i)X^{-1} = Xdiag(\lambda_i)Y^T = \sum_1^n \lambda_i x_i y_i^T,$$

где столбцы x_i матрицы X и строки y_i^T матрицы Y^T являются правыми и левыми собственными векторами A, нормированными так, что

$$y_i^T x_i = 1.$$

Следовательно,

$$A^s = Xdiag(\lambda_i^s)Y^T = \sum_1^n \lambda_i^s x_i y_i^T, \tag{8}$$

и если

$$|\lambda_1| = |\lambda_2| = ... = |\lambda_r| > |\lambda_r + 1| \geq ... \geq |\lambda_n|,$$

то главный член в правой части (8) есть $\sum_1^n \lambda_i^s x_i y_i^T$. Это основной результат, на котором основаны все методы этой главы.

Мы будем называть собственные значения $\lambda_1, \ldots, \lambda_r$ доминирующими собственными значениями, а соответствующие собственные векторы доминирующими собственными векторами. Наиболее часто r=1, и в этом случае главный член в ^{s} есть $\lambda_1^{s*} x_1 * y_1^T$.

Большинство практических способов основано на так называемом степенном методе вместе с приемами, позволяющими увеличить доминирование собственного значения наибольшего по модулю. Последние используют тот факт, что если p(A) и q(A) – полиномы от A, то

$$p(A)\{q(A)\}^{-1} = Xdiag\{p(\lambda_i)/q(\lambda_i)\}Y^T,$$

при условии, что q(A) — неособенная. Если мы обозначим

$$\mu_i = p(\lambda_i)/q(\lambda_i),$$

то целью является такой выбор полиномов p(A) и q(A), чтобы один из $|\mu_i|$ был много больше других.

Простейшее применение метода таково. Пусть u_0 произвольный вектор, и пусть последовательности векторов υ_s и u_s определяются уравнениями

$$\upsilon_{s+1} = Au_s, u_{s+1} = \upsilon_{s+1}/max(\upsilon_{s+1});$$

Здесь и в дальнейшем мы используем обозначение max(x) для максимального по модулю элемента вектора x. Очевидно, мы имеем

$$u_s = A^s u_0/max(A^s u_0),$$

и если мы положим, что

$$u_0 = \sum_1^n \alpha_i x_i,$$

то с точностью до нормирующего множителя, u_s имеет вид

$$\sum_1^n \alpha_i \lambda_i^s x_i = \lambda_i^s [\alpha_1 x_1 + \sum_2^n \alpha_i (\lambda_i/\lambda_1)^s x_i].$$

Если $|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq ... \geq |\lambda_n|$, то, предполагая $\alpha_1 neq 0$, получим

$$u_s \to x_1/max(x_1), max(\upsilon_s) \to \lambda_1.$$

Следовательно, этот процесс дает одновременно доминирующее собственное значение и соответствующий собственный вектор. Если $|\lambda_1/\lambda_2|$ близко к единице, то сходимость будет очень медленная.

Если существует несколько линейно независимых собственных векторов, соответствующих доминирующему собственному значению то это не

влияет на сходимость. Действительно, если

$$\lambda_1 = \lambda_2 = ... = \lambda_r, |\lambda_| > |\lambda_{r+1}| \geq ... \geq |\lambda_n|,$$

то

$$A^s u_0 = \lambda_1^s [\sum_1^r \alpha_i x_i + \sum_{r+1}^n \alpha_i (\lambda_i/\lambda_1)^s x_i] \sim \lambda_1^s \sum_1^r \alpha_i x_i.$$

Итерации поэтому сходятся к некоторому вектору, лежащему в подпространстве, натянутом на собственные векторы $x_1, \ldots, x_r$, причем, предел зависит от начального вектора u_0.

Данный метод не применяется для случая обобщенной собственной проблемы, так как имеет низкую скорость сходимости даже для случая одной матрицы $(A - \lambda I)$.

3.2 Метод обратных итераций

Это итеративный метод в численном анализе, который позволяет найти приближенный собственный вектор при наличии приближенного соответствующего собственного значения. Алгоритм очень близок к степенному методу (прямой итерации). Изначально он был разработан для вычисления резонансных частот в области структурной механики.

Для применения метода требуется наличие приближенного собственного значения μ и некого вектора b_0, который должен быть случайным или приближенным к соответствующему собственному вектору.

Алгоритм описывается итерациями следующего уравнения

$$b_{k+1} = \frac{(A - \mu I)^{-1} b_k}{C_k},$$

где C_k это константа, например, нормированный числитель, то есть

$$C_k = ||(A - \mu I)^{-1} b_k||$$

Получается, что на каждой итерации вектор b_k умножается на $(A - \mu I)^{-1}$ и нормализуется.

Чем ближе к реальному собственному значению выбрано μ, тем быстрее будет сходимость. Если же число μ выбрано неправильно, то сходимость будет очень низкой или даже сойтись к другим собственным значению и вектору.

Основную формулу можно переписать в следующий вид, если домножить уравнение на матрицу $(A - \mu I)$ слева

$$(A - \mu I)b_{k+1} = \frac{b_k}{C_k},$$

в такой форме будет легче соотнести уравнение для обобщенного случая (матричного пучка).

Для пучков матриц исходный метод подвергся модификации. Пучковый вариант метода обратных итераций для вычисления правого собственного вектора u_k для λ_k имеет предписание

$$x_{k(r+1)} = Bu_{k(r)};$$
$$u_{k(r+1)} = \frac{x_{k(r+1)}}{||x_{k(r+1)}||_2};$$
$$\lim_{r\to\infty} u_{k(r)} = u_k.$$

Для полиномиальных матриц (λ-матриц) имеем

$$D(\lambda_k)x_{k(r+1)} = \frac{d}{d\lambda}D(\lambda_k)u_{k(r)};$$
$$u_{k(r+1)} = \frac{x_{k(r+1)}}{||x_{k(r+1)}||_2};$$
$$\lim_{r\to\infty} u_{k(r)} = u_k.$$

Левый собственный вектор ищется как правый для транспонированной λ-матрицы или пучка матриц по тем же алгоритмам (4.6.2) или (4.6.3). Относительная погрешность вычислений собственных векторов, также как и собственных значений составляет величину не более 10^{-13}.

3.3 Уточнение собственных чисел через собственные вектора

Если для сходимости к λ_n используется сдвиг p, скорость сходимости, например, LR- и QR-алгоритмов определяется скоростью, с которой $[(\lambda_n - p)/(\lambda_n - 1 - p)]^s$ стремится к нулю. В этих процессах p выбрано приближением к λ_n. Мы можем получить столь же высокую скорость сходимости степенного метода, если будем итерировать с $(A\breve{}pI)^{-1}$, а не $(A\breve{}pI)$. С точностью до ошибок округления процесс имеет следующий вид:

$$\upsilon_{s+1} = (A - pI)^{-1} u_s, u_{s+1} = \upsilon_{s+1}/max(\upsilon_{s+1}).$$

Если

$$u_0 = \sum \alpha_i x_i,$$

то с точностью до нормирующего множителя имеем

$$u_s = \sum \alpha_i (\lambda_i - p)^{-s} x_i,$$

Следовательно, если $|\lambda_k - p| \ll |\lambda_i - p| (i \neq k)$, компоненты по $x_i (i \neq k)$ в u_s, быстро уменьшаются с ростом s.

Глава 4. Декомпозиционные методы

4.1 QR алгоритм

Данный метод решает собственную проблему: вычисление собственных чисел и векторов матрицы A. Был независимо разработан Френсисом и Кублановской в 1961 году. Ранее алгоритму предшествовал LR-алгоритм, который проводил разложение на нижнюю треугольную единичную матрицу L и верхнюю треугольную матрицу R.

Основная идея QR-алгоритма состоит в применении QR-разложения и записи матрицы как произведение ортогональной матрицы Q и верхней треугольной матрицы R. На каждом шаге производится данное разложение матрицы A, после чего получаем новую матрицу перемножение Q и R в обратном порядке

$$A_s = Q_s R_s, A_{s+1} = Q_s^H A_s Q_s = Q_s^H Q_s R_s Q_s = R_s Q_s,$$

На каждой стадии используется унитарное преобразование подобия. Если A_s неособенная, то эта факторизация по существу единственна, и она безусловно единственна, если мы возьмем диагональные элементы R_s вещественными и положительными. Если A_s — вещественная, то Q_s и R_s — вещественные. Эта факторизация имеет преимущество, что обращение в ноль ведущего главного минора A_s не вызывает нарушения алгоритма, как это происходит в LR-разложении.

Последовательные итерации удовлетворяют соотношениям

$$A_{s+1} = Q_s^H A_s Q_s = Q_s^H Q_{s-1}^H A_{s-1} Q_{s-1} Q_s = (Q_s^H \cdots Q_2^H Q_1^H) A_(Q_1 Q_2 \cdots Q_s),$$

что дает:

$$Q_1 Q_2 ... Q_s A_{s+1} = A_1 Q_1 Q_2 ... Q_s.$$

Поэтому все A_s унитарно подобны A_1 и если положить

$$Q_1 Q_2 ... Q_s = P_s, R_s R_{s-1} ... R_1 = U_s,$$

то

$$P_s U_s = (Q_1 \cdots Q_{s-1})(Q_s R_s)(R_{s-1} \cdots R_1) = (Q_1 \cdots Q_{s-1}) A_s (R_{s-1} \cdots R_1) = A_1 (Q_1 \cdots$$

Следовательно,

$$P_s U_s = A_1^s,$$

так что P_s и U_s дают соответствующую факторизацию A_1^s. Эта факторизация единственна, если диагональные элементы верхней треугольной матрицы положительны, и так будет для U_s, если это верно для всех R_s.

Через определенное количество итераций вещественная несимметричная матрица A сойдется к верхней треугольной форме (форме Шура), на диагонали которой будут находиться собственные числа. Если же A содержит комплексные собственные числа, то она сойдется к блочной верхней треугольной матрице, на диагонали которой будут блоки 1x1 и 2x2. Блоки первого порядка будут содержать действительные собственные числа, а блоки второго порядка – пары комплексно сопряженных собственных чисел.

В чистом виде каждая итерация алгоритма будет довольно затратной, поэтому в прикладных программах применяются такие улучшения как:

- Приведение матрицы A к форме Хессенберга (посредством отражений Хаусхолдера или вращений Гивенса);
- Применение одинарных и двойных сдвигов;
- Проведение неявных итераций;
- Дефляция (обнуление поддиагональных элементов).

4.2 Декомпозиция Шура (Schur decomposition) и QZ алгоритм

Пусть A это квадратная матрица, содержащая комплексные или/и действительные элементы. Тогда A можно представить как разложение Шура:

$$A = Q^* \cdot U \cdot Q,$$

где Q — унитарная, Q^* — её эрмитово-сопряжённая, а U — верхняя треугольная матрица, называемая комплексной формой Шура, которая содержит собственные значения A на диагонали. Данное разложение можно получить, например, QR-алгоритмом, рассмотренным выше.

Для случая пучков матриц ($Ax = \lambda Bx$) существует метод, являющийся аналогом разложения Шура и QR-метода для матриц – обобщенное разложение Шура или QZ-разложение. Оно выглядит следующим образом:

$$A = Q \cdot S \cdot Z; B = Q \cdot T \cdot Z,$$

где Q и Z — ортогональные матрицы, S — квази-верхняя треугольная и T — верхняя треугольная матрица.

В результате такого разложения матрица Q — содержит левые собственные вектора пучка, Z — содержит правые собственные вектора, а собственные числа λ_i пучка являются $\lambda_i = S_{i,i}/T_{i,i}$. Получены все обобщенные собственные числа и вектора.

4.3 JDQZ алгоритм

При $\lambda = \alpha/\beta$ обобщенная проблема ($Ax = \lambda Bx$) эквивалентна собственной проблеме

$$\beta A - \alpha B)x = 0, \tag{9}$$

и обозначает обобщенное собственное число матричной пары A, B как пары (α, β). Этот подход предпочитаем по той причине, что опустошения или перегрузки для $\lambda = \alpha/\beta$ в арифметике конечной точности могут возникать,

когда α и/или β ноли или близки к нолю. Даже в таких случаях пара (α, β) будет хорошо определена.

Частичная обобщенная форма Шура размерности k для пары матриц A, B это декомпозиция вида

$$AQ_k = Z_k R_k^A, BQ_k = Z_k R_k^B, \tag{10}$$

где Q_k и Z_k это ортогональные n*k матрицы R_k^A и R_k^B это верхний треугольные матрицы k*k. Столбцы q_i в Q_k являются обобщенными векторами Шура и относятся к паре $((\alpha_i, \beta_i), q_i)$. Пара $(\alpha_i, \beta_i) = (R_k^A(i,i), R_k^B(i,i))$ является обобщенной парой Шура. Из этого следует, что если $((\alpha, \beta), y)$ это обобщенная собственная пара (R_k^A, R_k^B), тогда $((\alpha, \beta), Q_k * y)$ это обобщенная собственная пара A, B.

Сейчас опишем метод Якоби-Дэвидсона для обобщенной собственной проблемы (9). Из отношения (10) мы заключаем, что

$$\beta_i A q_i - \alpha_i B q_i \perp z_i,$$

которое предлагает, что нужно следовать условию Петрова-Галеркина для построения сокращенной системы. На каждом шаге приближенный собственный вектор u выбирается из подпространства поиска span(V). Мы требуем, чтобы остаток от $(\eta * A * u \breve{} \zeta * B * u)$ был ортогонален к некоторому другому хорошо подобранному тестовому подпространству span(W)

$$\eta A u - \zeta B u \perp span(W). \tag{11}$$

Оба подпространства одной размерности, скажем, m. Уравнение (11) ведет к проецируемой собственной проблеме

$$(\eta W^* A V - \zeta W^* B V) s = 0. \tag{12}$$

Пучок $(\eta W^* A V - \zeta W^* B V)$ может быть преобразована с помощью QZ-алгоритма к обобщенной форме Шура (учитывая, что это проблема с размерностью m). Это приводит к ортогональным m*m матрицам S^R и S^L

и верхним треугольным m*m матрицам T^A и T^B, таким что

$$(S^L)^*(W^*AV)S^R = T^A, (S^L)^*(W^*BV)S^R = T^B. \quad (13)$$

Декомпозиция может быть пересортирована так, что первая колонка S^R и (1, 1)-ые вхождения T^A и T^B представляют искомое решение Петрова. С $s \equiv s_1^R \equiv S^R * e_1$ и $\zeta \equiv T_{1,1}^A$, $\eta \equiv T_{1,1}^B$ определяется вектор Петрова, как $u \equiv Vs$ для соответствующих обобщенных значений Петрова (ζ, η).

В текущем алгоритме строятся ортогональные матрицы V и W, такие что еще и $||u||_2 = 1$. С декомпозицией в (13), мы построим приближение частичной обобщенной формы Шура (см. (10): VS^R приближается к Q_k и WS^L приближается к соответствующему Z_k. Так как $span(Z_k) = span(AQ_k) = span(BQ_k)$ (см. (10)), это предполагает выбор W такого, что $span(W)$ совпадает со $span(\nu_0 AV + \mu_0 BV)$. С весами ν_0 и μ_0 можно повлиять на сходимость значений Петрова. Если нужно приближение собственных пар к собственным числам λ (близких к целевому значению τ), тогда выбор

$$\nu_0 = 1/\sqrt{1 + |\tau|^2}, \mu = -\tau\nu_0$$

является очень эффективным, особенно, если нужно вычислить собственные числа находящиеся внутри спектра $A - \lambda B$. Будем называть приближения Петрова для этого случая — гармоническими собственными парами Петрова. Уточняющее уравнение Якоби-Дэвидсона для компонента $t \perp u$ для пучка $\eta A \breve{} \zeta B$ становится

$$(I - \frac{pp^*}{p^*p})(\eta A - \zeta B)(I - uu^*)t = -r, \quad (14)$$

с $r \equiv \eta Au - \zeta Bu$ и $p \equiv \nu_0 Au + \mu_0 Bu$. Это можно показать так, что если (14) точно решено, то сходимость к обобщенной собственной проблеме будет квадратичной. Обычно, это уточняющее уравнение решается только приближенно, например, с (предобусловенным) итеративным решателем. Полученный вектор t используется для расширения v в V и $\nu_0 Au + \mu_0 Bu$ используется для расширения W. Для обоих пространств работа идет с

ортонормальными базисами. Следовательно, новые столбцы ортонормализованны по отношению к текущему базису с помощью модифицированного процесса ортогонализации Грэма—Шмидта.

Можно показать так, что с вышеуказанным выбором p и W,

$$p = W s_1^L = W S^L e_1.$$

Связь между частичной формой Шура для данной большой задачи и полной формой Шура для малой задачи (12) по правым векторам ($u = V s_1^R$) аналогична связи для левых векторов ($p = W s_1^L$).

Для ускорения алгоритма применяется дефляция и перезапуски.

Глава 5. Сравнение, анализ, вычислительный эксперимент

Данная глава посвящена сравнению эффективности нескольких декомпозиционных алгоритмов для решения обобщенной собственной проблемы (получения собственных чисел и векторов пучка матриц).

Сравнение будет проводится в среде математического пакета Matlab. Для сравнения были выбраны следующие алгоритмы:

1. qz() - является встроенным алгоритмом пакета Matlab, в старых версиях использовал процедуры из пакетов Linpack, Eispack и Lapack. Возвращает верхние треугольные и ортогональные матрицы разложения Шура, правые и левые собственные вектора.

2. eig() - это встроенный алгоритм пакета Matlab, который в старых версиях использовал процедуры из пакетов Linpack, Eispack и Lapack. Возвращает все собственные числа, правые и левые собственные вектора.

3. eigs() - в последней версии является встроенным методом Matlab. Позволяет настраивать количество возвращаемых собственных значений, начальное опорное $\lambda(\tau)$ и другие опции. Возвращает собственные числа и правые собственные вектора.

4. eigsUsingARPACK() - аналог метода eigs(), но из более старых версий, чем текущая. Использует алгоритмы пакета Arpack. Настройки и выходные переменные идентичны предыдущему пункту.

5. jdqz() - алгоритм, описанный в подглаве (4.3). Написан математиком по имени Gerard Sleijpen, являющимся одним из разработчиков данного метода.

Далее представлены таблицы, содержащие измерения времени рассчета собственной проблемы пучков разреженных матриц. В первом столбце название методов, в первой строке указана размерность каждого из рассчитываемых пучков. Величины указаны в секундах.

Таблица 1: Таблица сравнения алгоритмов

Метод	20*20	50*50	301*301

QZ	0,0215859	0,1117095	1,1031315
EIG	0,0136735	0,0988028	0,8839222
EIGS	0,0166411	0,0517071	0,6257575
eigsUsingARPACK	0,0144011	0,0389235	0,5720641
JDQZ	0,9978597	1,5801657	33,3992135

Таблица 2: Таблица сравнения алгоритмов

Метод	661*661	1020*1020	2904*2904
QZ	14,628623	58,1679036	1717,5291102
EIG	9,0730626	36,4840307	1165,6109484
EIGS	6,9167709	29,9464975	848,2294546
eigsUsingARPACK	6,6898750	30,7017752	836,4305394
JDQZ	644,3189726	1929,336203	–

По полученным результатам можно сделать вывод, что текущая реализация JDQZ сильно проигрывает функциям Matlab. А среди функций пакета Matlab наиболее эффективными являются EIGS и eigsUsingARPACK. Заметно и то, что алгоритмы QZ и EIG становятся все менее быстрыми с увеличением размерности матриц, в сравнении с EIGS и eigsUsingARPACK.

Заключение

В данной работе проведено исследование алгоритмов для решения полной собственной проблемы больших разреженных пучков матриц, получаемых в моделировании радиоэлектронных схем. Рассматривались интерполяционные, степенные и декомпозиционные методы.

Обнаружен новый способ применения собственных векторов линейной системы — выявление неустойчивых компонент при анализе устойчивости системы.

Было проведено сравнение скорости работы декомпозиционных алгоритмов на разреженных пучках и определение наиболее эффективных. На данный момент, такие алгоритмы применимы только для небольших схем.

Необходимо улучшать существующие алгоритмы, искать варианты оптимизации для работы с большими матрицами, чтобы внедрять их на практике в анализе устойчивости систем моделирования.

Список литературы

[1] Уилкинсон Дж. Х. «Алгебраическая проблема собственных значений». Редакторы: И. М. Овчинникова и Г. С. Росляков, Издательство «Наука», Главная редакция физико математической литературы — М., 1970г. — 564 с.

[2] В. Н. Кублановская, В. Н. Фаддеева «Вычислительные методы для решения обобщенной проблемы собственных значений». Тр. МИАН СССР, 1962г. — том 66, 147–165с.

[3] Гантмахер Ф. Р. «Теория матриц». Издательство «Наука» — М., 1967. — 576 с.

[4] Гридин В. Н., Михайлов В. Б., Шустерман Л. Б. «Численно-аналитическое моделирование радиоэлектронных схем». Отв. ред. Емельянова Е. В., Центр информ. технологий в проектировании РАН, Издательство «Наука» — М., 2008г. — 339 с.

[5] Ланкастер П. «Теория матриц». Пер. с англ.: Издательство «Наука», Главная редакция физико математической литературы — М., 1982г. — 272 с.

[6] David E. Muller «A Method for Solving Algebraic Equations Using an Automatic Computer». Published by: American Mathematical Society, Source: Mathematical Tables and Other Aids to Computation, Vol. 10, No. 56 (Oct., 1956), pp. 208–215.

[7] Moler C. B., Stewart G. W. «An algorithm for generalized matrix eigenvalue problems». The University Of Michigan, 1972, pp. 11–31.

[8] Gerard L.G. Sleijpen, Henk A. Van Der Vorst, and Ellen Meijerink «Efficient expansion of subspaces in the Jacobi-Davidson method for standard and generalized eigenproblems». ISSN 1068-9613, Electronic Transactions on Numerical Analysis. Volume 7, 1998, pp. 75-89.

[9] D. R. Fokkema, G. L. G. Sleijpen, and H. A. van der Vorst. «Jacobi-Davidson style QR and QZ algorithms for the partial reduction of matrix pencils». SIAM J. Sci. Comput., 20:94–125, 1998.