

# Численное решение в **Scilab** однородной задачи об изгибе балки методом Галеркина с конечно-элементной аппроксимацией

ревизия\*№2

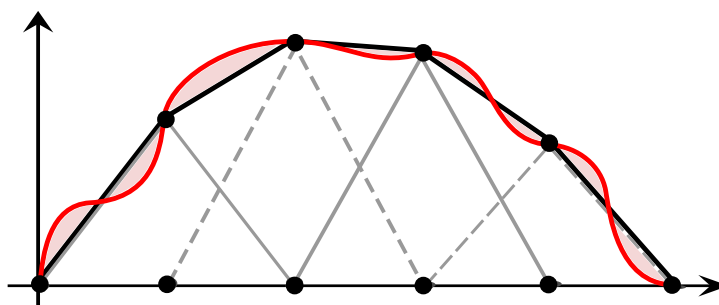
Дмитрий Мальков<sup>†</sup>

г. Комсомольск-на-Амуре

8 ноября 2014 г.

## Аннотация

Рассмотрено решение одномерной однородной задачи из теории упругости методом Галеркина с интерполяцией на конечных элементах. Приводится программный код решения задачи на ЭВМ в математическом комплексе общего назначения **Scilab**.



---

\*статьи время от времени пересматриваются и исправляются, скачивайте свежайшие версии с домашней страницы автора

<sup>†</sup>**Email:** maldmitrix@gmail.com, **сайт:** <http://maldmitrix.github.io>

# Содержание

<b>Предисловие</b>	<b>3</b>
<b>1 Интерполяция сплайнами</b>	<b>4</b>
1.1 Быстрое введение в интерполяцию . . . . .	4
1.2 Интерполяция Эрмита . . . . .	6
<b>2 Численное решение задачи о статических прогибах балки</b>	<b>8</b>
2.1 Уравнение Эйлера-Бернулли изгиба балки . . . . .	8
2.2 Решение однородной задачи методом Галеркина с конечно-элементной аппроксимацией . . . . .	9
<b>Литература</b>	<b>12</b>
<b>Приложение А</b>	<b>13</b>

## Предисловие

Определить, понимается численный метод или нет, очень легко. Если не возникает проблем при попытке сесть за компьютер и реализовать в математическом программном комплексе общего назначения, например, метод Бубнова-Галеркина (далее просто Галеркина), то этот метод понимается. К математическим комплексам общего назначения относятся такие программные системы, как **Matlab**, **Scilab**, **GNU Octave**, **Maple**, **Mathematica**, **MathCad**, **Maxima** и подобные им. Изучение сложных численных методов в таких универсальных программах, а не в специализированных, дает глубокое понимание их сути.

В приложении к этой статье приведен пример программирования решения задачи теории упругости методом Галеркина-МКЭ (через дефис обычно указывается метод аппроксимации). Сама задача и алгоритм решения приведены в последнем разделе. Алгоритм запрограммирован в комплексе **Scilab** пятой версии. Однако код можно легко переделать, чтобы он запустился в **Matlab** или **GNU Octave**.

# 1 Интерполяция сплайнами

## 1.1 Быстрое введение в интерполяцию

*Аппроксимация* – замена одних математических объектов другими, в том или ином смысле близкими к исходным. *Аппроксимацией (приближением) функции  $f(x)$*  называется нахождение такой функции  $\tilde{f}(x)$  (аппроксимирующей функции), которая была бы близка к заданной. Критерии близости функций  $f(x)$  и  $\tilde{f}(x)$  могут быть различные.

В том случае, когда приближение строится на дискретном наборе точек, аппроксимацию называют *точечной* или *дискретной*. В том случае, когда аппроксимация проводится на непрерывном множестве точек (отрезке), аппроксимация называется *непрерывной* или *интегральной*. Примером такой аппроксимации может служить разложение функции в ряд Тейлора, то есть замена некоторой функции степенным многочленом.

Мы будем рассматривать точечную аппроксимацию. При этом функция  $f(x)$  как правило неизвестна, а связь между параметрами  $x$  и  $y$  задается в виде некоторой таблицы  $\{x_i, y_i\}$ . Это означает, что дискретному множеству значений аргумента  $\{x_i\}$  поставлено в соответствие множество значений функции  $\{y_i\}$  ( $i = 0, 1, \dots, n$ ). Эти значения – либо результаты расчетов, либо экспериментальные данные. На практике же (например, для визуализации) могут понадобиться значения величины  $y$  и в других точках, отличных от узлов  $x_i$ . Однако получить эти значения можно лишь путем очень сложных расчетов или проведением дорогостоящих экспериментов. В подобных случаях, оптимальным, с точки зрения экономии времени и средств, является использование имеющихся табличных данных для приближенного вычисления искомого параметра  $y$  при любом значении определяющего параметра  $x$  (в рамках некоторого интервала), поскольку точная связь  $y = f(x)$  неизвестна или использование ее в расчетах затруднительно.

Общая погрешность аппроксимирующей функции может быть выражена как сумма локальных погрешностей в точках с координатами  $x_i$ :  $E = \sum_i e_i$ , где  $e_i = |\tilde{f}(x_i) - f(x_i)|$ . В общем случае при аппроксимации  $0 \leq e_i \leq \varepsilon$ . В случае, если  $\varepsilon = 0$ , то есть налагается условие строгого совпадения значений функций  $\tilde{f}(x)$  и  $f(x)$  в заданных точках  $x_i$ , то данный вид аппроксимации называется *интерполяцией*.

При интерполяции  $\tilde{f}(x_i) = f(x_i)$ , что автоматически подразумевает наличие известных  $\{x_i, y_i\}$ , для некоторого определенного интервала  $[x_0, x_n]$ . В случае, если требуется получить аппроксимацию функции за пределами известного интервала, то данный вид аппроксимации называется *экстраполяцией*.  $x_i$ , для которых даны  $y_i$ , называются *узлами интерполяции* или *опорными точками*.

Интерполяция бывает *глобальной* –  $\tilde{f}(x)$  проходит через все точки

заданного интервала  $[x_0, x_n]$ , либо *локальной (кусочной)* –  $f(x)$  на указанном интервале интерполируется несколькими  $\tilde{f}_1(x), \tilde{f}_2(x), \dots, \tilde{f}_k(x)$ .

Известны три типа интерполяции: полиномиальная, тригонометрическая, экспоненциальная.

*Линейная интерполяция* – простейший вид локальной полиномиальной интерполяции – замена  $f(x)$  множеством линейных функций  $\tilde{f}_1(x), \tilde{f}_2(x), \dots, \tilde{f}_k(x)$ , каждая из которых соединяет лишь две точки. Если заданы две точки  $f(x_0) = y_0$  и  $f(x_1) = y_1$ , то их соединяет отрезок прямой, уравнение которой есть полином первой степени

$$\tilde{f}(x) = \frac{y_1 - y_0}{x_1 - x_0}x + y_0.$$

Пусть в интервале  $[a, b]$  заданы  $n + 1$  опорных точек  $a \leq x_0 < x_1 < x_2 < \dots < x_n \leq b$ , а так же  $n + 1$  действительных чисел  $y_i$  ( $i = 0, 1, \dots, n$ ). Тогда в качестве глобальной интерполяционной функции  $\tilde{f}(x)$  можно найти многочлен степени не больше  $n$ , такой, что  $\tilde{f}(x_i) = y_i$ . Всегда существует *единственный* интерполяционный многочлен. Но однозначно определенный многочлен может быть представлен в различных видах.

Пусть кривые интерполирующих функций задаются параметрически по  $t$ . *Непрерывность* интерполирующих функций при кусочной интерполяции:

- Непрерывность нулевого порядка по параметру,  $C_0$  – означает, что кривые встречаются, то есть  $\tilde{f}_{k-1}(x_p) = \tilde{f}_k(x_p)$ .
- Непрерывность первого порядка по параметру,  $C_1$  – означает, что первые производные по параметру ( $t$ ) двух кривых одинаковы в точке пересечения (стыковки), то есть  $\tilde{f}'_{k-1}(x_p) = \tilde{f}'_k(x_p)$ .
- Непрерывность второго порядка по параметру,  $C_2$  – означает, что первая и вторая производные по параметру ( $t$ ) двух кривых одинаковы в точке пересечения (стыковки), то есть  $\tilde{f}'_{k-1}(x_p) = \tilde{f}'_k(x_p)$  и  $\tilde{f}''_{k-1}(x_p) = \tilde{f}''_k(x_p)$ .

Для гладкой интерполяции узлов изобрели сплайны. *Сплайны* ( $k$ -сплайны) – кусочные полиномы степени  $k$  с непрерывной производной степени  $k - 1$  в точках соединения сегментов. Иными словами, сплайнами называется набор функций, который вместе с несколькими производными этих функций непрерывен на отрезке  $[a, b]$ , а на каждом частном интервале этого отрезка  $[x_i, x_i + 1]$  в отдельности является некоторым многочленом невысокой степени. В настоящее время чаще всего применяют *кубический сплайн*, то есть на каждом локальном интервале функция является полиномом 3-го порядка.

## 1.2 Интерполяция Эрмита

Примечательная особенность Эрмитовой интерполяции в том, что кроме значений  $y_i$  интерполируемой функции  $f(x)$  в точках  $x_i$ , также учитываются и ее первые производные в этих точках. Этот дополнительный учет добавляет два уравнения в систему, определяющую параметры интерполирующей функции. Чтобы получить уникальные значения этих параметров, число независимых уравнений в системе должно быть равно числу параметров, поэтому нужно увеличить число параметров. Два дополнительных параметра дают возможность построить гладкий сплайн, у которого при переходе с сегмента на сегмент не так сильно заметна смена полинома (выполняется условие непрерывности  $C_1$ ). Если значения первых производных неизвестны (а чаще всего это именно так), то есть разные способы их найти, взяв в расчет узлы соседних сегментов.

Если даны две точки  $\{x_0, x_1\}$ , значения интерполируемой функции в них  $\{f(x_0), f(x_1)\}$  и значения первых производных  $\{f'(x_0), f'(x_1)\}$ , то кубический полином Эрмита можно записать как линейную комбинацию четырех базисных функций  $h_i$ :

$$H_3(x) = h_1 f(x_0) + h_2 f'(x_0) + h_3 f(x_1) + h_4 f'(x_1),$$

$$h_1 = \left(1 + 2 \frac{x - x_0}{x_1 - x_0}\right) \left(\frac{x_1 - x}{x_1 - x_0}\right)^2, \quad h_2 = (x - x_0) \left(\frac{x_1 - x}{x_1 - x_0}\right)^2,$$
$$h_3 = \left(1 + 2 \frac{x_1 - x}{x_1 - x_0}\right) \left(\frac{x_0 - x}{x_0 - x_1}\right)^2, \quad h_4 = (x - x_1) \left(\frac{x_0 - x}{x_0 - x_1}\right)^2.$$

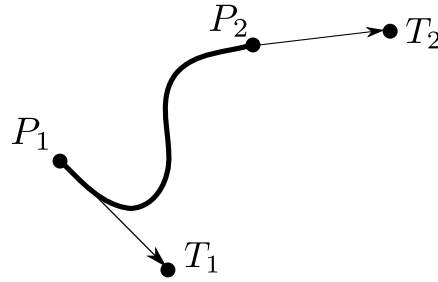


Рис. 1: Чтобы определить кубический полином, нужно задать стартовую и конечную точки  $P_1, P_2$ , направления и скорости кривой в этих точках  $T_1, T_2$  (каким образом кривая покидает точку  $P_1$  и встречается точку  $P_2$ )

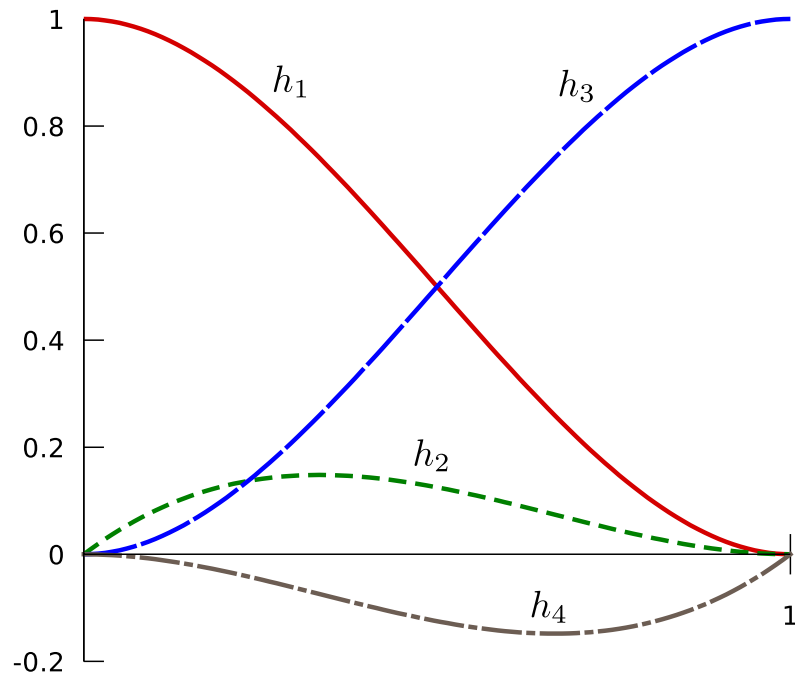


Рис. 2: Графики четырех Эрмитовых базисных функций  $h_i$ , линейная комбинация которых образует кубический интерполяционный Эрмитов полином для каждого отдельного сегмента кубического сплайна

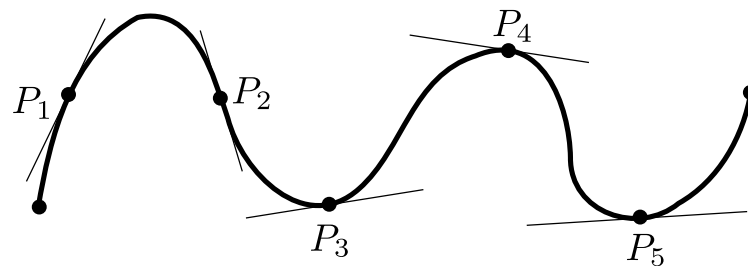


Рис. 3: При переходе с одного кубического сплайна на другой, первая производная не меняется.  $P_i$  – точки стыков сегментов

## 2 Численное решение задачи о статических прогибах балки

### 2.1 Уравнение Эйлера-Бернулли изгиба балки

Балочная теория Эйлера-Бернулли, называемая классической теорией, чаще всего применяется в расчетах благодаря ее простоте и удовлетворительным погрешностям во многих задачах. В этой теории поперечные прогибы балки записываются в виде обыкновенного дифференциального уравнения четвертого порядка

$$\frac{d^2}{dx^2} \left[ r(x) \frac{d^2 u(x)}{dx^2} \right] = f(x, u), \quad 0 \leq x \leq \ell,$$

подчиненного условиям на границе

$$u(0) = a_0, \quad \frac{d^2 u(0)}{dx^2} = b_0, \quad u(\ell) = a_\ell, \quad \frac{d^2 u(\ell)}{dx^2} = b_\ell.$$

Функция  $r(x) = E \cdot I$  – изгибная жесткость балки,  $E$  – модуль упругости при растяжении,  $I$  – осевой момент инерции поперечного сечения балки относительно ее нейтральной оси.  $u(x)$  – это прогибы балки,  $f(x, u)$  – распределенная поперечная нагрузка, зависящая в том числе и от величины прогиба в каждой точке. В случае линейности,  $f(x, u) = q(x) \cdot u(x) + p(x)$ , и тогда уравнение можно переписать как

$$\frac{d^2}{dx^2} \left[ r(x) \frac{d^2 u(x)}{dx^2} \right] = q(x)u(x) + p(x), \quad 0 \leq x \leq \ell,$$

где  $q(x)$  – коэффициент эластичности основания (грунта) и  $p(x)$  – распределение поперечных к балке активных сил.

Если балка не лежит на основании, а поддерживается только на ее концах или на одном конце (консольная балка), то  $q(x) = 0$  решение  $u(x)$  описывает прогибы балки при распределенной активной нагрузке  $p(x)$ .

Для случаев, когда заданы простые функции  $f(x, u)$  и  $r(x)$ , точное решение обыкновенного дифференциального уравнения может быть легко найдено стандартными путями. Но в более сложных случаях найти точное решение затруднительно, и тогда стоит применять численные методы для нахождения приближенного решения. Покажем далее, как применить метод Галеркина с методом Конечных Элементов в случае, когда  $r(x) = E \cdot I$  – константа,  $q(x) = 0$ . Задача будет иметь вид

$$EI \frac{d^4 u(x)}{dx^4} = p(x), \quad 0 \leq x \leq \ell,$$



$$u(0) = a_0, \quad \frac{d^2 u(0)}{dx^2} = b_0, \quad u(\ell) = a_\ell, \quad \frac{d^2 u(\ell)}{dx^2} = b_\ell.$$

## 2.2 Решение однородной задачи методом Галеркина с конечно-элементной аппроксимацией

Сначала решим задачу глобально, используя глобальные базисные и весовые функции, целиком покрывающие область от 0 до  $\ell$ .

Запишем исходное дифференциальное уравнение в слабой форме, умножив невязку на весовую функцию  $w(x)$  и взяв интеграл по области задачи. Интегрирование по частям позволит понизить порядок дифференцирования функции  $u$ :

$$\begin{aligned} \int_0^\ell \left[ EI \frac{d^4 u(x)}{dx^4} - p(x) \right] w(x) dx &= EI \frac{d^3 u(x)}{dx^3} w(x) \Big|_0^\ell - \\ &- EI \frac{d^2 u(x)}{dx^2} \frac{dw(x)}{dx} \Big|_0^\ell + \int_0^\ell \left[ EI \frac{d^2 w(x)}{dx^2} \frac{d^2 u(x)}{dx^2} - p(x) w(x) \right] dx = 0. \end{aligned}$$

Теперь нужно выбрать подходящие базисные (аппроксимирующие) функции  $u_i$ . Заметим, что наибольший порядок производной от  $u(x)$  в слабой форме равен трем. Выберем базисные функции трижды дифференцируемые. Базисные кубические интерполяционные полиномы Эрмита  $h_i$  подойдут. После подстановки  $x_0 = 0$  и  $x_1 = \ell$  в формулы базисных кубических полиномов Эрмита, они принимают вид

$$\begin{aligned} h_1 &= 1 - 3 \left( \frac{x}{\ell} \right)^2 + 2 \left( \frac{x}{\ell} \right)^3, & h_2 &= x \left( 1 - \frac{x}{\ell} \right)^2, \\ h_3 &= 3 \left( \frac{x}{\ell} \right)^2 - 2 \left( \frac{x}{\ell} \right)^3, & h_4 &= x \left[ \left( \frac{x}{\ell} \right)^2 - \frac{x}{\ell} \right]. \end{aligned}$$

Как принято в методе взвешенных невязок, представим пробную функцию  $\tilde{u}$  в форме разложения по данным базисным функциям  $\tilde{u} = \sum_{j=1}^4 u_j h_j$ . Используя метод Галеркина, мы должны взять весовые функции такие же, как и базисные:  $w_i = h_i$ . Подставим в уравнение слабой формы весовые функции. Получим:

$$\begin{aligned} \int_0^\ell \left[ EI \frac{d^4 u(x)}{dx^4} - p(x) \right] w(x) dx &= EI \frac{d^3 u(x)}{dx^3} h_i(x) \Big|_0^\ell - \\ &- EI \frac{d^2 u(x)}{dx^2} \frac{dh_i(x)}{dx} \Big|_0^\ell + \int_0^\ell \left[ EI \frac{d^2 h_i(x)}{dx^2} \frac{d^2 u(x)}{dx^2} - p(x) h_i(x) \right] dx = 0. \end{aligned}$$

Матрица жесткости будет состоять из компонентов

$$K_{ij} = EI \int_0^\ell \frac{d^2 h_i}{dx^2} \frac{d^2 h_j}{dx^2} dx.$$

Вектор-столбец внешних сил будет состоять из компонентов

$$f_i = \int_0^\ell p(x) h_i dx.$$

Найдем, например, первый компонент матрицы жесткости:

$$\begin{aligned} K_{11} &= EI \int_0^\ell \frac{d^2 h_1}{dx^2} \frac{d^2 h_1}{dx^2} dx = EI \int_0^\ell \frac{1}{\ell^3} (12x - 6\ell) \frac{1}{\ell^3} (12x - 6\ell) dx = \\ &= \frac{EI}{\ell^6} \int_0^\ell 144x^2 - 144x\ell + 36\ell^2 dx = \\ &= \frac{EI}{\ell^6} (48x^3 - 72x\ell + 36x\ell) \Big|_0^\ell = \frac{EI}{\ell^6} (12\ell^3) = \frac{12EI}{\ell^3}. \end{aligned}$$

Все остальные компоненты найдутся аналогично, и матрица жесткости будет иметь вид:

$$K_{ij} = \frac{EI}{\ell^3} \begin{bmatrix} 12 & 6\ell & -12 & 6\ell \\ 6\ell & 4\ell^2 & -6\ell & 2\ell^2 \\ -12 & -6\ell & 12 & -6\ell \\ 6\ell & 2\ell^2 & -6\ell & 4\ell^2 \end{bmatrix}.$$

Аналогично найдем первый компонент вектор-столбца сил, для простоты считая, что распределенная сила не меняет свою интенсивность:

$$f_i = \int_0^\ell p \left( 1 - \frac{3x^2}{\ell^2} + \frac{2x^3}{\ell^3} \right) dx = p \left( x - \frac{x^3}{\ell^2} + \frac{x^4}{2\ell^3} \right) \Big|_0^\ell = \frac{p\ell}{2}.$$

Полный вектор-столбец сил:

$$f_i = \frac{p\ell}{2} \begin{bmatrix} 1 \\ 6\ell \\ 1 \\ -6\ell \end{bmatrix}.$$

И можно записать матричное уравнение:

$$\frac{EI}{\ell^3} \begin{bmatrix} 12 & 6\ell & -12 & 6\ell \\ 6\ell & 4\ell^2 & -6\ell & 2\ell^2 \\ -12 & -6\ell & 12 & -6\ell \\ 6\ell & 2\ell^2 & -6\ell & 4\ell^2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \frac{p\ell}{2} \begin{bmatrix} 1 \\ 6\ell \\ 1 \\ -6\ell \end{bmatrix}$$

Решив это матричное уравнение, найдем вектор-столбец перемещений.

Теперь нужно разбить область на  $m$  элементов, каждый из которых имеет длину  $h = \frac{\ell}{m}$  (не путайте с полиномами Эрмита  $h_i$ ). Для каждого элемента составить свое локальное матричное уравнение. Далее ассемблировать элементные уравнения в единое глобальное матричное уравнение с порядком, равным числу степеней свободы, а их у нас по две на каждый узел (узлов всего  $m + 1$  штук): значение перемещения в узле и значение первой производной от перемещения. Далее подчинить глобальное матричное уравнение граничным условиям: скрыть от вычислений и результатов столбцы и строки, соответствующие определенным и неизменяемым степеням свободы. Если нет перемещения в узле, не меняется первая степень свободы. Если при этом нет и поворота сечения, то не меняется вторая степень свободы в этом же узле. Наконец, решить матричное уравнение процедурами линейной алгебры.

## Список литературы

- [1] Bruce A. Finlayson The method of weighted residuals and variational principles. – Academic press, inc., New York, 1972.
- [2] S Rao. Gunakala, D.M.G. Comissiong. A Finite Element Solution of the Beam Equation via MATLAB. – International Journal of Applied Science and Technology, Vol. 2 No. 8; October 2012, pp 80-88.
- [3] Иванов В.Н. Вариационные принципы и методы решения задач теории упругости: Учеб. пособие – М.: Изд-во РУДН, 2004. – 176 с.: ил.
- [4] Бабенко К.И. Основы численного анализа. – М.: Наука, 1986.

## Приложение А

Код в программе Scilab (после небольших поправок код можно запустить в Matlab или в GNU Octave) для расчета методом Галеркина с конечно-элементной аппроксимацией прогибов балки с разными условиями на концах. Балка имеет длину 1, всюду постоянную изгибную жесткость равную 1 и постоянное поперечное сечение. Нагрузка - поперечная, равномерно распределенная. Балку представляем как последовательность равноудаленных узлов. Результат – вертикальные перемещения узлов – представляем как точечный график смещенных узлов. «Чисто декоративно» соединяем узлы отрезками прямых линий, чтобы нагляднее представить непрерывную балку.

```
m = 100; // число конечных элементов
P = -100; // нагрузка

// Массив из m+1 равномерно распределенных точек на интервале [0,1]
nodeCoordinates = linspace(0, 1, m+1)';
L = max(nodeCoordinates); // Наибольшая координата узлов, т.е. L=1
numberNodes = size(nodeCoordinates,1);
xx = nodeCoordinates;
E = 1; I = 1; EI = E*I; // Изгибная жесткость балки
GDof = 2*numberNodes; // Глобальное число степеней свобод
U = zeros(GDof,1);
force = zeros(GDof,1); // Глобальный столбец сил
stiffness = zeros(GDof,GDof); // Глобальная матрица жесткости
displacements = zeros(GDof,1); // Глобальный столбец прогибов

// Нумерация узлов каждого элемента (для первого индексы: [1, 2])
for i = 1:m
    elementNodes(i,1) = i;
    elementNodes(i,2) = i+1;
end

for e=1:m // Пробегаем все элементы
    indice=elementNodes(e,:); // Номера узлов элемента (2 штуки)
    // Номера степеней свобод, принадлежащих элементу (4 штуки)
    elementDof = [ 2*(indice(1)-1)+1 2*(indice(2)-1) ...
                  2*(indice(2)-1)+1 2*(indice(2)-1)+2];
    h = xx(indice(2)) - xx(indice(1)); // Длина элемента
    // Набиваем локальный столбец сил каждого элемента
    f1 = (P*h/2)*[1 6*h 1 -6*h]';
    force(elementDof)=force(elementDof)+f1;
    // Набиваем локальную матрицу жесткости каждого элемента
    k1=(EI/h^3)*[ 12 6*h -12 6*h ;...
                 6*h 4*h^2 -6*h 2*h^2 ;...
                 -12 -6*h 12 -6*h ;...
                 6*h 2*h^2 -6*h 4*h^2 ]';
    stiffness(elementDof,elementDof)=stiffness(elementDof,elementDof)+k1;
end
```

```

// Граничные условия: оба конца заземлены
fixedNodeU = [1 2*m+1];
fixedNodeV = [2 2*m+2];
prescribedDof = [fixedNodeU;fixedNodeV];
activeDof = setdiff([1:GDof]',[prescribedDof]);
U = stiffness(activeDof,activeDof) \ force(activeDof);
displacements = zeros(GDof,1);
displacements(activeDof) = U;
U = displacements(1:2:2*numberNodes);
plot(nodeCoordinates,U,'b:'); // синий штрих чере две точки

```

```

// Граничные условия: оба конца свободно оперты
fixedNodeU = [1 2*m+1];
fixedNodeV = [];
prescribedDof = [fixedNodeU;fixedNodeV];
activeDof = setdiff([1:GDof]',[prescribedDof]);
U = stiffness(activeDof,activeDof) \ force(activeDof);
displacements = zeros(GDof,1);
displacements(activeDof) = U;
U = displacements(1:2:2*numberNodes);
plot(nodeCoordinates,U,'r--'); // красный штрих

```

```

// Граничные условия: левый конец заземлен
// правый конец свободно оперт
fixedNodeU = [1 2*m+1];
fixedNodeV = [2 2*m+1];
prescribedDof = [fixedNodeU;fixedNodeV];
activeDof = setdiff([1:GDof]',[prescribedDof]);
U = stiffness(activeDof,activeDof) \ force(activeDof);
displacements = zeros(GDof,1);
displacements(activeDof) = U;
U = displacements(1:2:2*numberNodes);
plot(nodeCoordinates,U,'k-'); // черная сплошная линия

```

```

// Граничные условия: левая половина балки заземлена,
// правая половина вся свободна
fixedNodeU = [1:m];
fixedNodeV = [1:m];
prescribedDof = [fixedNodeU;fixedNodeV];
activeDof = setdiff([1:GDof]',[prescribedDof]);
U = stiffness(activeDof,activeDof) \ force(activeDof);
displacements = zeros(GDof,1);
displacements(activeDof) = U;
U = displacements(1:2:2*numberNodes);
plot(nodeCoordinates,U,'g-'); // зеленая сплошная линия

```

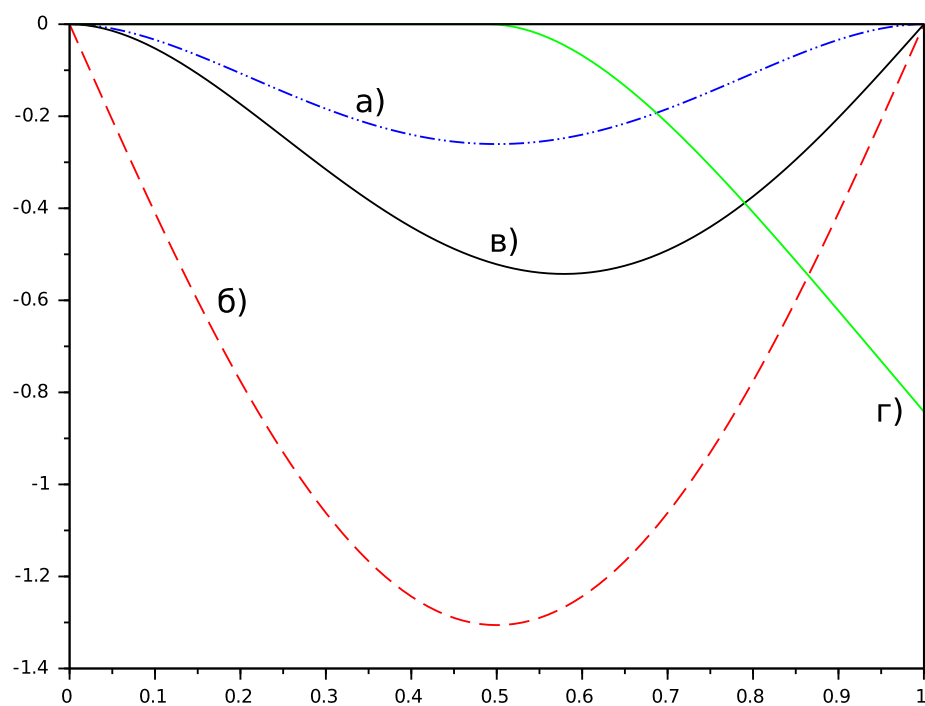


Рис. 4: графики прогибов балки при одинаковой нагрузке, выданные приведенным **Scilab**-скриптом: а) концы защемлены; б) концы свободно опираются; в) левый конец защемлен, правый конец свободно оперт; г) левая половина балки вся защемлена, остальная часть вся свободна