

# 4. Получение решения линейной модели (построение матричной экспоненты и интеграла от неё)

Андрей Бареков      Ярослав Пылаев  
По лекциям Устинова С.М.

June 26, 2020

# 1 Получение решения линейной модели

Рассмотрим линейную динамическую модель

$$\frac{dx}{dt} = Ax + b \quad (1)$$

Для решения системы (1) можно было использовать стандартные программы решения дифференциальных уравнений (например, RKF45), однако для линейных систем можно предложить специальный алгоритм, который эффективно работает в т.ч. для жёстких систем.

Точное решение системы (1) имеет вид<sup>1</sup>:

$$x(t) = e^{At}x_0 + \int_0^t e^{A\tau} d\tau \times b \quad (2)$$

Воспользоваться разложением в ряд для построения  $e^{At}$  затруднительно, т.к. нужно изменять  $t$ . Использование формулы Лагранжа-Сильвестра даже для матрицы средней размерности требует расчёта собственных значений и векторов и использования громоздких формул.

Поэтому на практике используют следующий подход:

Пусть  $H$  - желаемый шаг визуализации решения (шаг печати)<sup>2</sup>. Запишем решение (2) в точке  $t + H$ :

$$x(t + H) = e^{A(t+H)}x_0 + \int_0^{t+H} e^{A\tau} d\tau \times b \quad (3)$$

Умножим 2 на  $e^{AH}$  и вычтем из уравнения 3:

$$\tau + H = \tau^*$$

$$\begin{aligned} x(t + H) - e^{AH}x(t) &= \int_0^{t+H} e^{A\tau} d\tau \times b - \int_0^t e^{A(\tau+H)} d\tau \times b \\ &= \int_0^{t+H} e^{A\tau} d\tau \times b - \int_H^{t+H} e^{A\tau^*} d\tau^* \times b \\ &= \int_0^H e^{A\tau} d\tau \times b \end{aligned}$$

$$x(t + H) = e^{AH}x(t) + \int_0^H e^{A\tau} d\tau \times b \quad (4)$$

Теперь необходимо **однократно** построить матричную экспоненту  $e^{AH}$  и её интеграл, а затем решить уравнение (4) пошаговым методом.

Матрицу  $e^{AH}$  можно было бы построить разложением в ряд, однако, по

---

<sup>1</sup> Из курса вычислительной математики, №18

<sup>2</sup> В.м.,  $h_{print}$  из RKF45, №26

второй теореме о матричных функциях<sup>3</sup>:

$$A = U \Lambda U^{-1}$$

$$f(A) = U f(\Lambda) U^{-1}$$

$$e^{AH} = U \begin{pmatrix} f(\lambda_1) & 0 & \dots & 0 \\ 0 & f(\lambda_2) & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & f(\lambda_n) \end{pmatrix} U^{-1} = U \begin{pmatrix} e^{\lambda_1 H} & 0 & \dots & 0 \\ 0 & e^{\lambda_2 H} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & e^{\lambda_n H} \end{pmatrix} U^{-1}$$

Использование разложения в ряд  $e^{AH}$  сводится к разложению в ряд скалярных функций  $f(\lambda_k) = e^{\lambda_k H}$

**Пример:**

$$e^{-0.1} = 1 - 0.1 + \frac{0.01}{2} - \frac{0.001}{6} + \frac{0.0001}{24} + \dots$$

$$e^{-10} = 1 - 10 + \frac{100}{2} - \frac{1000}{6} + \frac{10^4}{24} + \dots$$

При разложении в ряд малое число слагаемых будет только при  $|\lambda_k H| \approx 1$ , а если система жёсткая, то это требует очень маленького шага

**Проблема:**

Умею строить разложение в ряд  $e^{Ah}$  при малых  $h$  с небольшим кол-вом слагаемых. Как построить  $e^{AH}$  при большом  $H$ ?

Получить  $e^{AH}$  можно, используя процедуру удвоения шага.

$$x(t+H) = \varphi(AH)x(t) + g(H), \quad g(H) = \int_0^H e^{A\tau} d\tau \times b \quad (4^*)$$

$$\varphi(2h) = \varphi(h) \times \varphi(h) \quad (5)$$

Используя формулу (5)  $S$  раз, получаем  $\varphi(AH)$ ,  $H = 2^S \times h$

Для вектора  $g(H)$  также существует процедура удвоения шага:

$$\tau = \tau^* + h$$

$$g(2h) = \int_0^{2h} e^{A\tau} d\tau \times b = \int_0^h e^{A\tau} d\tau \times b + \int_h^{2h} e^{A\tau} d\tau \times b$$

$$= g(h) + e^{Ah} \int_0^h e^{A\tau^*} d\tau^* \times b = (E + \varphi(Ah))g(h)$$

$$g(2h) = (E + \varphi(Ah))g(h) \quad (6)$$

$\varphi(Ah)$  и  $g(h)$  находим разложением в ряд с небольшим количеством слагаемых:

$$\varphi(Ah) = E + Ah + \frac{A^2 h^2}{2!} + \frac{A^3 h^3}{3!} + \dots [4] \quad (7)$$

<sup>3</sup>В.м., №17

<sup>4</sup>ряд для экспоненты

$$\varphi(Ah) = hE + \frac{Ah^2}{2!} + \frac{A^2h^3}{3!} + \dots [5] \quad (8)$$

Вместо  $\lambda_k$  можно использовать норму -  $h\|A\| \approx 1$

#### Детализируем алгоритм по шагам:

1. Задаёмся желаемым значением  $H$ , выбираем минимальное целое значение  $S$ , такое, что  $h = \frac{H}{2^S} < \frac{1}{\|A\|}$
2. Для  $h$  строим  $\varphi(Ah)$  и  $g(h)$  по формулам (7) и (8) с небольшим количеством слагаемых
3.  $S$  раз используем формулы (5) и (6) удвоения шага и получаем матрицу  $\varphi(AH)$  и  $g(H)$
4. Решаем уравнение (4\*) пошаговым методом.

Входные параметры программы следующие:

- $N$  - размер матрицы
- $A$  - сама матрица
- $B$  - вектор  $b$
- $X$  - начальное приближение  $x$
- $H$  - желаемый шаг визуализации решения

В списке параметров отсутствуют  $\varepsilon_A$  и  $\varepsilon_R$ . В отсутствие ошибок округления погрешность возникает только в формулах (7) и (8), которые могут быть реализованы с любой степенью точности. Степень жёсткости системы проявляется только в величине  $S$ .

## 2 Некоторые свойства собственных векторов матриц $A$ и $A^T$

$$\begin{aligned} A &\rightarrow \lambda_k, U_k \\ A^T &\rightarrow \lambda_k, V_k \\ AU_k &= \lambda_k U_k \\ A^T V_i &= \lambda_i V_i \end{aligned} \quad (1)$$

---

<sup>5</sup> получается интегрированием ряда для экспоненты

$$V_i^T A = \lambda_i V_i^T \quad (2)$$

Умножим уравнение (1) слева на  $V_i^T$ , а уравнение (2) справа на  $U_k$ .  
Вычитаем результаты:

$$0 = (\lambda_k - \lambda_i) V_i^T U_k$$

Если  $i \neq k$ , то:

$$V_i^T \times U_k = 0 \quad (*!)$$

Если  $V_i$  и  $U_k$  вещественные, то собственные векторы для  $A^T$  и  $A$ , относящиеся к различным  $\lambda_k$ , ортогональные.

## 2.1 Формула Лагранжа-Сильвестра

Формула (\*!) позволяет записать формулу Лагранжа-Сильвестра в более компактном виде.

$$f(A) = \sum_{k=1}^N T_k f(\lambda_k), T_k = \frac{(A - \lambda_1 E) \dots (A - \lambda_{k-1} E)(A - \lambda_{k+1} E) \dots (A - \lambda_n E)}{(\lambda_k - \lambda_1) \dots (\lambda_k - \lambda_{k-1})(\lambda_k - \lambda_{k+1}) \dots (\lambda_k - \lambda_n)}.$$

Умножим  $T_k \times U_i$  ( $k \neq i$ ). Скобку  $(A - \lambda_i E)$  в числителе поставим последней.

$$(A - \lambda_i E) U_i = 0 \Rightarrow T_k U_i = 0, k \neq i$$

Умножим  $T_k \times U_k$ . Последовательным умножением скобок на  $U_k$  убеждаемся в том, что  $T_k U_k = U_k$ .

Пусть  $z$  - первая строчка матрицы  $T_k$ . Разложим этот вектор по векторам  $V_i^T$ :

$$z = \alpha_1 V_1^T + \alpha_2 V_2^T + \dots + \alpha_N V_N^T$$

Умножаем  $z \times U_i$  ( $k \neq i$ ).  $\alpha_i V_i^T U_i = 0, i \neq k$  Таким образом, первая строчка  $T_k$  содержит только одно слагаемое -  $\alpha_k V_k^T$ . Аналогичный вид будут иметь все остальные строки матрицы  $T_k$ . Матрица  $T_k$  имеет ранг 1, и её можно записать в виде:

$$T_k = C_k \times V_k^T$$

, где вектор  $C_k$  содержит множители каждой строки. Нормируем векторы  $V_k$  и  $U_k$  так, чтобы  $V_k^T U_k = 1$ .

Тогда  $T_k U_k = C_k \underbrace{V_k^T U_k}_1 = C_k = U_k$ .

Таким образом матрица  $T_k$  имеет вид  $T_k = U_k V_k^T$ , и формула Лагранжа-Сильвестра приобретает следующий вид:

$$f(A) = \sum_{k=1}^N U_k V_k^T f(\lambda_k)$$