МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ
Кафедра методов оптимального управления

МЕТОДЫ УПРАВЛЕНИЯ ПО ПРОГНОЗИРУЮЩЕЙ МОДЕЛИ НА ОСНОВЕ ДАННЫХ

Отчёт по практике

Кулешова Владислава Вячеславовича
студента 4 курса,
специальность «прикладная математика»

Научный руководитель:
канд. физ.-мат. наук
доцент Н.М. Дмитрук

Минск, 2021

ОГЛАВЛЕНИЕ

С.

ГЛАВА 1

ТЕОРИЯ

1.1 Основные предположения

Пусть $\mathbb{I}_{[a,b]}$ – множество целых чисел на отрезке $[a, b]$. Для вектора x и положительно определённой матрицы $P = P^T > 0$ запишем $\|x\|_P = \sqrt{x^T P x}$. Далее определим максимальное и минимальное собственные значения матрицы P $\lambda_{\min}(P)$ и $\lambda_{\max}(P)$.

Для матриц $P_1 = P_1^T, P_2 = P_2^T$ запишем,

$$\lambda_{\min}(P_1, P_2) = \min\{\lambda_{\min}(P_1), \lambda_{\min}(P_2)\},$$

$$\lambda_{\max}(P_1, P_2) = \max\{\lambda_{\max}(P_1), \lambda_{\max}(P_2)\}.$$

Также $\|x\|_2, \|x\|_1, \|x\|_\infty$ – Евклидовы, ℓ_1 и ℓ_∞ нормы x, соответственно. Для $\delta > 0$, мы определим $\mathbb{B}_\delta = \{x \in \mathbb{R}^n \mid \|x\|_2 \leq \delta\}$.

Из последовательности $\{x_k\}_{k=0}^{N-1}$ составим матрицу Ганкеля

$$H_L(x) = \begin{bmatrix} x_0 & x_1 & \dots & x_{N-L} \\ x_1 & x_2 & \dots & x_{N-L+1} \\ \vdots & & \ddots & \vdots \\ x_{L-1} & x_L & \dots & x_{N-1} \end{bmatrix},$$

$$x_{[a,b]} = \begin{bmatrix} x_a \\ \vdots \\ x_b \end{bmatrix}$$

Определение 1 $\{x_k\}_{k=0}^{N-1}$ с $x_k \in \mathbb{R}^n$ постоянно возбуждающая порядка L, если $\operatorname{rank}(H_L(x)) = nL$.

Цель паботы – управление неизвестной линейной стационарной системой G с порядком n, с m входами и p выходами.

Определение 2 Последовательность $\{u_k, y_k\}_{k=0}^{N-1}$ – траектория линейной стационарной системы G, если существуют начальное состояние $\overline{x} \in \mathbb{R}^n$,

а также соответсвующая траектория $\{x_k\}_{k=0}^{N}$ такая, что

$$x_{k+1} = Ax_{k+1} + Bu_k;\ x_0 = \overline{x},$$

$$y_k = Cx_k + Du_k,\ k = 0, \ldots, N-1,$$

(A, B, C, D) – минимальные реализация G.

Теорема 1.1 Предположим, что $\{u_k^d, y_k^d\}_{k=0}^{N-1}$ – траектория линейной стационарной системы G, где u – постоянно возбуждающее управление порядка $L + n$.
Тогда $\{\overline{u}_k, \overline{y}_k\}_{k=0}^{L-1}$ – траектория G, тогда и только тогда, когда $\exists \alpha \in \mathbb{R}^{N-L+1}$ такое, что

$$\begin{bmatrix} H_L(u^d) \\ H_L(y^d) \end{bmatrix} \alpha = \begin{bmatrix} \overline{u} \\ \overline{y} \end{bmatrix}. \tag{1.1}$$

Определение 3 Пара $(u^s, y^s) \in \mathbb{R}^{m+p}$ – положение равновесия линейной стационарной системой G, если последовательность $\{\overline{u}_k, \overline{y}_k\}_{k=0}^{n-1}$ с $(\overline{u}_k, \overline{y}_k) = (u^s, y^s)\ \forall k \in \mathbb{I}[0, n-1]$ – траектория G.

Для равновесия (u^s, y^s) мы определим u_n^s, y_n^s как столбец векторов содержащий n раз u^s и y^s, соответственно. Предположим, что система подчиняется ограничениям, $u_t \in \mathbb{U} \subseteq \mathbb{R}^m, y_t \in \mathbb{Y} \subseteq \mathbb{R}^p\ \forall t \geq 0$, и предположим, что $(u^s, y^s) \in int(\mathbb{U} \times \mathbb{Y})$. $\{u_k^d, y_k^d\}_{k=0}^{N-1}$ – априори измеряемые траектории длинной N, использующиеся в (1.1). Прогнозируемые входные и выходные траектории в момент времени t в течение некоторого горизонта прогнозирования L записываются как $\{\overline{u}_k(t), \overline{y}_k(t)\}_{k=-n}^{L-1}$. Обратим внимание, что индексы времени начинаются с $k = -n$, так как последние n входов и выходов будут использоваться для вызова уникального начального состояния в момент времени t. Кроме того, вход с обратной связью, состояние в некоторой минимальной реализации и выход в момент времени t обозначаются как u_t, x_t, y_t, соответственно.

1.2 Задача с терминальными ограничениями-равенствами

В этом пункте рассмотрим простое терминальное ограничение, которое может быть включено непосредственно в структуру управления данными MPC.

$$J_L^*(u_{[t-n,t-1]}, y_{[t-n,t-1]}) = \min_{\alpha(t)\ \overline{u}(t),\overline{y}(t)} \sum_{k=0}^{L-1} \ell(\overline{u}_k(t), \overline{y}_k(t)) \tag{1.2}$$

$$\begin{bmatrix} \overline{u}_{[-n,-L-1]}(t) \\ \overline{y}_{[-n,-L-1]}(t) \end{bmatrix} = \begin{bmatrix} H_{L+n}(u^d) \\ H_{L+n}(y^d) \end{bmatrix} \alpha(t), \tag{1.3}$$

$$\begin{bmatrix} \overline{u}_{[-n,-1]}(t) \\ \overline{y}_{[-n,-1]}(t) \end{bmatrix} = \begin{bmatrix} u_{[t-n,t-1]} \\ y_{[t-n,t-1]} \end{bmatrix}, \tag{1.4}$$

$$\begin{bmatrix} \overline{u}_{[L-n,L-1]}(t) \\ \overline{y}_{[L-n,L-1]}(t) \end{bmatrix} = \begin{bmatrix} u_n^s \\ y_n^s \end{bmatrix}, \tag{1.5}$$

$$\overline{u}_k(t) \in \mathbb{U}, \overline{y}_k(t) \in \mathbb{Y}, k \in \mathbb{I}_{[0,L-1]}. \tag{1.6}$$

Терминальное ограничение-равенство (1.5) подразумевает, что $\overline{x}_L(t)$, который является внутренним состоянием, предсказанным на L шагов вперед, соответствующей предсказанной траекторией, выравнивается с постоянным состоянием x^s, соответствующим (u^s, y^s), то есть $\overline{x}_L(t) = x^s$ в любой минимальной реализации. В то время как задача требует, чтобы (u^s, y^s) было равновесием неизвестной системы по определению (3), это требование может быть отброшено, когда (u^s, y^s) заменено искусственным равновесием, которое также оптимизируется онлайн. Расширение представленной схемы MPC до такой настройки является предметом будущей работы. Как и в стандартном MPC, задача решена в виде отступающего горизонта, который обобщен в алгоритме (1).

Алгоритм 1 (Схема управления на основе MPC)

1. В момент времени t взять прошлые n измерений $u_{[t-n,t-1]}, y_{[t-n,t-1]}$ и решить задачу.

2. Взять за управление $u_t = \overline{u}_0^*(t)$

3. Установить $t = t + 1$ и вернуться к пункту 1).

1.3 Робастная схема MPC с терминальными ограничениями

На практике выходной сигнал неизвестной системы G обычно является неточным. Это означает, что сложенные матрицы Ганкеля, зависящие от данных, в 1.1 не покрывают пространство траекторий системы точно и, следовательно, выходные траектории не могут быть точно предсказаны. Более того, измерения выходного сигнала с зашумлением входят в начальные условия в задаче (1.2-1.6), что еще больше ухудшает точность прогноза.

В данном разделе будет предполагаться, что выходные сигналы с ограниченным аддитивным шумом находятся в изначально доступных данных $\widetilde{y}_k^d = y_k^d + \varepsilon_k^d$ и в измерениях $\widetilde{y}_k = y_k + \varepsilon_k$. Мы не делаем никаких предположений о природе шума, но требуем, чтобы он был ограничен как $\|\varepsilon_k^d\|_\infty \leq \overline{\varepsilon}$ и $\|\varepsilon_k\|_\infty \leq \overline{\varepsilon}$ для некоторого $\overline{\varepsilon}$. Ключевой идеей для учета зашумленных измерений является ослабление ограничения равенства (1.3). При зашумленной начальной траектории $(u_{[t-n,t-1]}, \widetilde{y}_{[t-n,t-1]})$ длины n, предлагается следующая надежная модификация (1.2-1.6).

$$J_L^*(u_{[t-n,t-1]}, \widetilde{y}_{[t-n,t-1]}) = \min_{\alpha(t),\overline{u}(t),\overline{y}(t),\sigma(t)} \sum_{k=0}^{L-1} \ell(\overline{u}_k(t), \overline{y}_k(t)) + \lambda_\alpha \overline{\varepsilon} \|\alpha(t)\|_2^2 + \lambda_\sigma \|\sigma(t)\|_2^2 \tag{1.7}$$

$$\begin{bmatrix} \overline{u}(t) \\ \overline{y}(t) + \sigma(t) \end{bmatrix} = \begin{bmatrix} H_{L+n}(u^d) \\ H_{L+n}(\widetilde{y}^d) \end{bmatrix} \alpha(t), \tag{1.8}$$

$$\begin{bmatrix} \overline{u}_{[-n,-1]}(t) \\ \overline{y}_{[-n,-1]}(t) \end{bmatrix} = \begin{bmatrix} u_{[t-n,t-1]} \\ y_{[t-n,t-1]} \end{bmatrix}, \tag{1.9}$$

$$\begin{bmatrix} \overline{u}_{[L-n,L-1]}(t) \\ \overline{y}_{[L-n,L-1]}(t) \end{bmatrix} = \begin{bmatrix} u_n^s \\ y_n^s \end{bmatrix}, \overline{u}_k \in \mathbb{U}, \tag{1.10}$$

$$\|\sigma_k(t)\|_\infty \leq \overline{\varepsilon}(1 + \|\alpha(t)\|_1), k \in \mathbb{I}_{[0,L-1]}. \tag{1.11}$$

По сравнению с задачей (1.2-1.6) траектория выходных данных $\widetilde{y}^d$ и начальный выходной сигнал $\widetilde{y}_{[t-n,t-1]}$, полученные из онлайн измерений, были заменены их зашумленными аналогами. Также были добавлены следующие элементы:

1. Вспомогательная переменная σ для олнайн измерений $\widetilde{y}_{[t-n,t-1]}$ и для зашумленных данных $\widetilde{y}^d$, используемых для предсказаний.

2. Квадратичная норма для σ и α с весами $\lambda_\alpha \overline{\varepsilon}, \lambda_\sigma > 0$, норма α зависит от уровня шума.

1.4 n-Шаговая схема управления на основе MPC

В этой главе мы рассматриваем управление по замкнутому контуру, полученное в результате применения (1.7-1.11) в n-шаговой схеме MPC. Чтобы быть более точным, мы рассматриваем сценарий, в котором сразу после решения (1.7-1.11) первые n вычисленных входных сигналов применяются к системе. После этого горизонт сдвигается на n шагов, прежде чем вся схема повторяется.

Алгоритм 2 (n-Шаговая схема управления на основе MPC)

1. В момент времени t взять прошлые n измерений $\{u_{[t-n,t-1]}, \widetilde{y}_{[t-n,t-1]}\}$ и решить задачу (1.7)-(1.11).

2. Взять за управление $u_{[t,t+n-1]} = \overline{u}^*_{[t,n-1]}(t)$

3. Установить $t = t + n$ и вернуться к пункту 1).

Задача (1.7-1.11) - это строго выпуклая квадратичная задача и она может быть эффективно решена. Однако ограничение на σ в (1.11) невыпукло из-за зависимости правой части от $\|\alpha(t)\|_1$, что затрудняет эффективную реализацию (1.7-1.11).(1.11) требуется для доказательства рекурсивной выполнимости и практической экспоненциальной устойчивости. Однако его можно заменить (выпуклым) ограничением $\|\sigma_k(t)\|_\infty \leq c \cdot \overline{\varepsilon}$ для достаточно большой постоянной $c > 0$ сохранением тех же теоретических гарантий. Как правило, больший выбор c увеличивает область притяжения, но также увеличивает размер экспоненциально устойчивого множества, к которому сходится замкнутый контур. Кроме того, ограничение (1.11) может быть реализовано неявно, выбирая λ_σ достаточно большим. В примерах моделирования было замечено, что ограничение (1.11) обычно выполняется (для достаточно большого выбора λ_σ), не применяя его явно в задаче оптимизации, и, таким образом, в большинстве случаев им можно пренебречь при онлайн-оптимизации.

ГЛАВА 2

ПРАКТИКА

2.1 Пример

Рассмотрим пример, который описывает систему из 4 сообщающихся резервуаров [3]. Это хорошо изученная в теории управления и в частности в теории MPC система. Она описывает достаточно медленный и устойчивый динамический процесс. Однако, известно также, что при выборе короткого горизонта управления система может стать неустойчивой. Линеаризованная динамика системы в дискретном времени описывается уравнениями

$$x_{k+1} = \begin{bmatrix} 0.921 & 0 & 0.041 & 0 \\ 0 & 0.918 & 0 & 0.033 \\ 0 & 0 & 0.924 & 0 \\ 0 & 0 & 0 & 0.937 \end{bmatrix} x_k + \begin{bmatrix} 0.017 & 0.001 \\ 0.001 & 0.023 \\ 0 & 0.061 \\ 0.072 & 0 \end{bmatrix} u_k,$$

$$y_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} x_k.$$

Целью управления будет являться отслеживание заданного значения системы

$$(u^s, y^s) = \left(\begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 0.65 \\ 0.77 \end{bmatrix} \right).$$

Предположим, что системные матрицы неизвестны, но доступна одна траектория входа-выхода $\{u_k^d, y_k^d\}_{k=0}^{N-1}$ длины $N = 400$, которая генерируется путем

равномерной выборки u_k^d из $[-1,1]^2$. Горизонт прогнозирования установим в $L = 30$, и следующие параметры $Q = 3 \cdot E_2$, $R = 10^{-4} \cdot E_2$, $\lambda_\sigma = 1000$, $\lambda_\alpha \overline{\varepsilon} = 0.1$.

2.2 Программная реализация алгоритмов

Рассмотрим способ реализации алгоритма (2) на языке Matlab. Сперва зададим матрицу и вектор ограничений-равенств для quadprog.

```
for i = 1:4:(L + n) * 4
    Aeq(i, i) = 1;
    Aeq(i + 1, i + 1) = 1;
    Aeq(i + 2, i + 2) = 1;
    Aeq(i + 3, i + 3) = 1;
    hIndex = fix(i / 4) + 1;
    Aeq(i, alphaRange) = -uHankel(hIndex, :, 1);
    Aeq(i + 1, alphaRange) = -uHankel(hIndex, :, 2);
    Aeq(i + 2, alphaRange) = -yHankel(hIndex, :, 1);
    Aeq(i + 3, alphaRange) = -yHankel(hIndex, :, 2);

    Aeq(i + 2, sigmaShift + sigmaIndex) = 1;
    Aeq(i + 3, sigmaShift + sigmaIndex + 1) = 1;

    sigmaIndex = sigmaIndex + 2;
end
```

Сведём задачу к минимизации функции

$$\min_x \frac{1}{2} x^T H x + f^T,$$

где H и f задаются следующим образом.

```
quadH = zeros(((L + n) * 4)  + (N - (L + n) + 1) + (L * 4) + (L + n) * 2);
quadF = zeros(1, ((L + n) * 4)  + (N - (L + n) + 1) + (L * 4) + (L + n) * 2);
for i = alphaShift+1:alphaRange(end)
    quadH(i,i)=lambdaAlphaEps;
end
for i = sigmaShift+1:sigmaRange(end)
    quadH(i,i)=lambdaSigma;
end
for i = n * 4 + 1:4:(L + n) * 4
    quadH(i, i) = coeffR;
```

```
    quadH(i + 1, i + 1) = coeffR;
    quadH(i + 2, i + 2) = coeffQ;
    quadH(i + 3, i + 3) = coeffQ;
    for j = steadyFirstIndex:4:numel(quadH(1, :))
        quadH(i, j) = -2 * uSteady(1) * coeffR;
        quadH(i + 1, j + 1) = -2 * uSteady(2) * coeffR;
        quadH(i + 2, j + 2) = -2 * ySteady(1) * coeffQ;
        quadH(i + 3, j + 3) = -2 * ySteady(2) * coeffQ;
    end
end
squares = L * 4 * ...
          (coeffR * uSteady(1)^2 + coeffR * uSteady(2)^2 ...
          + coeffQ * ySteady(1)^2 + coeffQ * ySteady(2)^2);
quadF(1, steadyFirstIndex) = squares;
quadH = quadH * 2;
```

После начнём итерационный процесс от 0 до N. В начале которого будем вычеслять вектор ограничений.

```
        Aeq(i + 2, sigmaRange) = 0;
        Aeq(i + 3, sigmaRange) = 0;

        beq(i) = uRes(j, 1);
        beq(i + 1) = uRes(j, 2);
        beq(i + 2) = yRes(j, 1);
        beq(i + 3) = yRes(j, 2);
        j = j + 1;
```

Зададим терминальные ограничения.

```
    for i = L * 4 + 1:4:(L + n) * 4
        Aeq(i + 2, sigmaRange) = 0;
        Aeq(i + 3, sigmaRange) = 0;

        beq(i) = uSteady(1, 1);
        beq(i + 1) = uSteady(2, 1);
        beq(i + 2) = ySteady(1, 1);
        beq(i + 3) = ySteady(2, 1);
    end
```

Решим задачу (1.7 - 1.11) с помощью quadprog.

```
    options = optimoptions('quadprog', ...
                           'MaxIter', 10000, ....
                           'TolFun', 1e-15, ...
```

```
                        'TolX', 1e-15);
[res, value] = quadprog(quadH, quadF, [], [], ...
                        Aeq, beq, [], [], [], options);
```

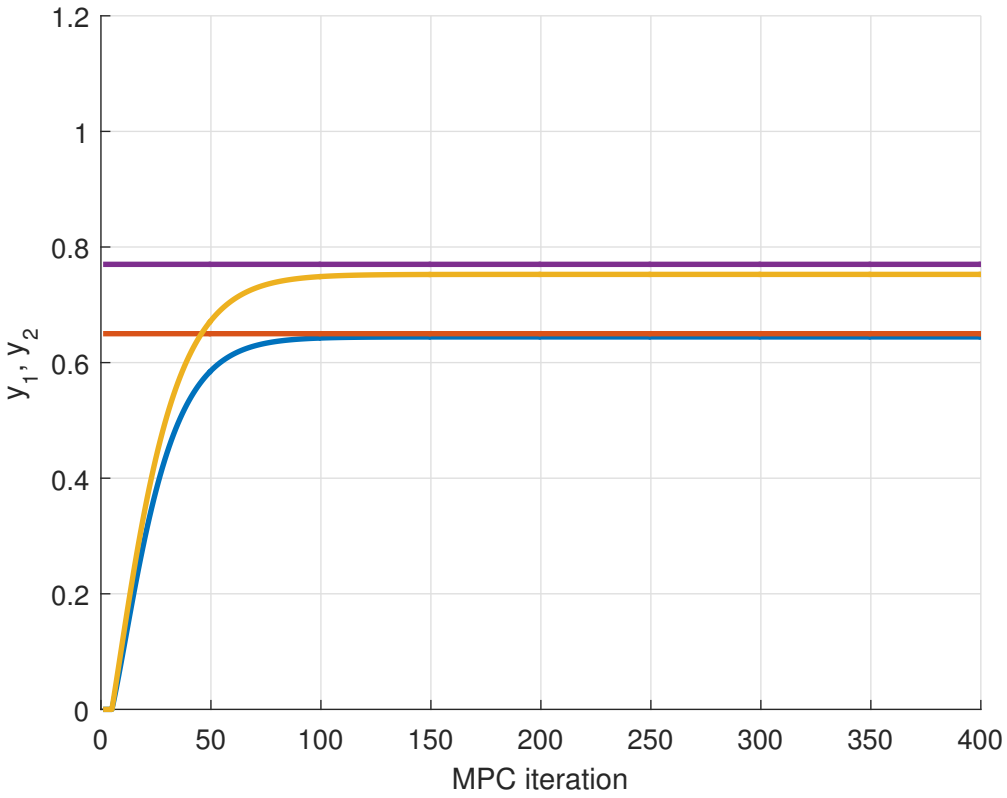
И запишем результат.

```
j = timeIndex - n;
for i = 1:4:(L + n) * 4
    uRes(j, 1) = res(i);
    uRes(j, 2) = res(i + 1);
    [dsResY, dsResX] = dynamicSystemFunc(uRes(j,:)', xRes(j,:)', A, B, C);
     yRes(j,:) = dsResY;
    j = j + 1;
    xRes(j,:) = dsResX';
end
```

2.3 Результаты

Ниже приведены графики результатов программ, описанных в прошлом пункте. На которых прямые – соответсвующее значения y_1^s, y_2^s.

Результаты программной реализации алгоритма (2):


1.2
1
0.8
0.6
0.4
0.2
0
y_1, y_2
0
50
100
150
200
250
300
350
400
MPC iteration

ЗАКЛЮЧЕНИЕ

В данной работе описанны основные определения теории управления по прогнозирующей модели, проанализированы схемы управления по прогнозирующей модели и показаны способы решений простейшей прогнозирующей задачи и задачи с терминальными ограничениями-равенствами,в которых для прогнозирования используются только прошлые измеренные данные без какого-либо предварительного шага идентификации системы. Также показано, что замкнутый контур в схеме управления по прогнозирующей модели рекурсивно допустим и практически экспоненциально устойчив.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1 Rawlings, J.B. Model Predictive Control: Theory and Design / J.B. Rawlings, D.Q. Mayne. – Madison: Nob Hill Publishing, 2009. – 576 p.

2 Методы оптимизации. Учебное пособие / В.В. Альсевич [и др.] – Мн.: «Четыре четверти», 2011.

3 Berberich, J. A trajectory-based framework for data-driven system analysis and control / J. Berberich, F. Allg?wer // arXiv, 2019 – 6p. (preprint arXiv:1903.10723).

4 Yang, H. A data-driven predictive controller design based on reduced hankel matrix / H. Yang, S. Li // Asian Control Conference: proc. of the 10th ACC – 2015. – 7p.

5 Coulson, J. Data-enabled predictive control: in the shallows of the DeePC / J.Coulson, J. Lygeros, F. D?rfler // European Control Conference: proc of the 18thECC. – 2019. – P. 307-312.