初心者向けテキスト

モデル予測制御

京都大学大学院工学研究科化学工学専攻
プロセスシステム工学研究室
加納　学

1997 年 1 月 第 1 版作成

［注意事項］
自由に利用していただいて結構ですが、本資料の間違いなどによって生じた不利益などに対して、著者は一切責任を負いません。勿論、間違いの指摘やアドバイスは歓迎します。

目 次

第1章　モデル予測制御の基礎

1.1　モデル予測制御とは

あるプロセスを制御したい場合、どのように操作量を決定すればよいだろうか。この問いに対する答えとして様々な制御方法が存在するが、中でもモデル予測制御は最も直感的な方法であると言える。ここでは、モデル予測制御がどのようにして操作量を決定しているのかを簡単に述べよう。

制御とは被制御量が設定値に一致するように操作量を変化させることである。この簡単な表現から次の結論を導くことができる。すなわち、被制御量が設定値に一致するように、操作量を決定すればよい。実際、プロセスの動的モデルが存在すれば、そのモデルに基づいて未来の被制御量の変化を予測できるので、被制御量と設定値が一致するような操作量を求めることは可能である。これこそが**モデル予測制御** (MPC; Model Predictive Control) の神髄である。

しかし、このような漠然とした表現のままでは計算機に実装することはできない。この概念を制御アルゴリズムとして実現するためには、以下の各項目を決定しなければならない。

- プロセスモデル
- 被制御量の予測式
- 操作量を決定するための評価関数

これらについて順に検討していこう。

1.2　プロセスモデル

MPC ではプロセスの動的モデルに基づいて被制御量 (出力) の予測を行うため、モデルの精度がそのまま制御性能の優劣となって現れる。しかし、精密な物理モデルの構築は極めて困難であるため、一般に利用されているモデルは、インパルス応答 (FIR; Finite Impulse Response) モデルおよびステップ応答 (FSR; Finite Step Response) モデルである。最近の市販ソフトウェアには、積分プロセスや不安定プロセスを取り扱うために、伝達関数 (TF; Transfer Function) モデルや ARX(Auto-Regressive eXogenous input) モデルを用いているものもある。また、理論面からの研究では、状態空間 (SS; State Space) モデルが広く利用されている。この理由としては、現代制御理論の成果を適用できること、多変数系への拡張が容易であることなどが挙げられる。ここでは、基本となるインパルス応答モデルおよびステップ応答モデルについて述べる。

1.2.1　インパルス応答モデル

MPC では離散時間モデルを用いるので、Fig.1.1 に示すようなパルス状入力に対する応答をインパルス応答と呼ぶ。パルス状入力は大きさ 1 で 1 ステップ間加えられるものとし、インパルス応答の各サンプリング時刻における値を $\{h_i\}$ と表すことにする。厳密にはインパルス応答係数 $\{h_i\}$ が有限ステップの間にゼロとなるとは限らないが、現実にはあるステップ以降はゼロとみなしても問題ない。そこで、$s+1$ ステップ

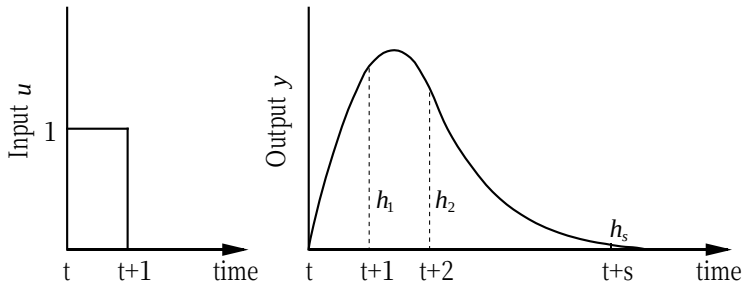

Fig. 1.1　インパルス応答

以降のインパルス応答係数はゼロ、すなわち

$$h_i \quad = \quad 0 \qquad \text{for } i > s \tag{1.1}$$

とする。なお、T_d ステップの無駄時間が存在する場合には、

$$h_i \quad = \quad 0 \qquad \text{for } i = 1, 2, \cdots, T_d \tag{1.2}$$

となる。

プロセスが線形であると仮定すると、時刻 t で大きさ $u(t)$ のパルス状入力をプロセスに加える場合、この入力に対する時刻 $t+j$ での出力の大きさは

$$y_M(t+j) \quad = \quad h_j u(t) \tag{1.3}$$

で与えられる。これを重ね合わせの原理を用いて一般的に表現することにより、次の**インパルス応答モデル**を得る。

$$\begin{aligned} y_M(t+j) \quad &= \quad \sum_{k=1}^{\infty} h_k u(t+j-k) \\ &= \quad \sum_{k=1}^{s} h_k u(t+j-k) \end{aligned} \tag{1.4}$$

あるいは

$$y_M(t) \quad = \quad \sum_{k=1}^{s} h_k u(t-k) \tag{1.5}$$

この式は、遅延演算子 d を用いて

$$\begin{aligned} y_M(t) \quad &= \quad \sum_{k=1}^{s} h_k d^k u(t) \\ &= \quad dG_M(d)u(t) \end{aligned} \tag{1.6}$$

と表すこともできる。なお、遅延演算子 d は

$$x(t-1) \quad = \quad d\,x(t) \tag{1.7}$$

で定義される。

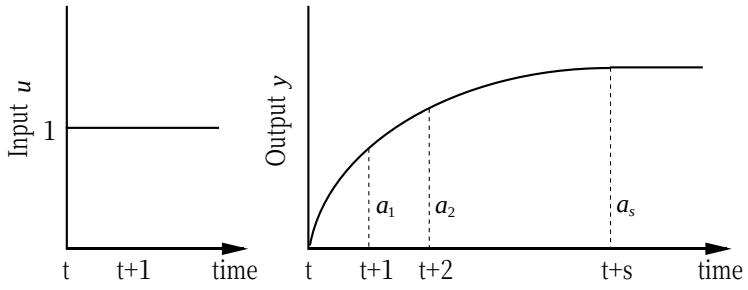

Fig. 1.2　ステップ応答

1.2.2　ステップ応答モデル

Fig.1.2 に示すような大きさ 1 のステップ状入力に対する応答を考える。ステップ応答の各サンプリング時刻における値を $\{a_i\}$ と表すことにする。厳密にはステップ応答係数 $\{a_i\}$ が有限ステップの間に一定値になるとは限らないが、現実にはあるステップ以降は一定とみなしても問題ない。そこで、s ステップ以降のステップ応答係数は一定、すなわち

$$a_i \quad = \quad a_s \qquad \text{for } i \geq s \tag{1.8}$$

とする。なお、T_d ステップの無駄時間が存在する場合には、

$$a_i \quad = \quad 0 \qquad \text{for } i = 1, 2, \cdots, T_d \tag{1.9}$$

となる。

プロセスが線形であると仮定すると、時刻 t で大きさ $\Delta u(t)$ のステップ状入力をプロセスに加える場合、この入力に対する時刻 $t+j$ での出力の大きさは

$$y_M(t+j) \quad = \quad a_j \Delta u(t) \tag{1.10}$$

で与えられる。これを重ね合わせの原理を用いて一般的に表現することにより、次の**ステップ応答モデル**を得る。

$$y_M(t+j) \quad = \quad \sum_{k=1}^{\infty} a_k \Delta u(t+j-k) \tag{1.11}$$

あるいは

$$y_M(t) \quad = \quad \sum_{k=1}^{\infty} a_k \Delta u(t-k) \tag{1.12}$$

この式は、遅延演算子 d を用いて

$$\begin{aligned} y_M(t) \quad &= \quad \sum_{k=1}^{\infty} a_k d^k \Delta u(t) \\ &= \quad dG_M(d)\Delta u(t) \end{aligned} \tag{1.13}$$

と表すこともできる。本来はインパルス応答モデルとステップ応答モデルとで異なる記号を用いるべきであるが、特に誤解の恐れがないかぎり、共に $G_M(d)$ を用いることにする。

1.3 被制御量の予測式

モデルがプロセスを完全に表現している場合、Sec.1.2 で述べたモデルを用いて計算される出力の推定値は実測値と一致し、予測値 $y_P(t+j)$ を

$$y_P(t+j) \quad = \quad y_M(t+j) \tag{1.14}$$

としてもよい。しかし、現実には、外乱やモデル化誤差のために推定値と実測値とは一致しない。そこで、何らかの補正が必要になるが、通常、現時刻 t での実測値と推定値との差が補正項として用いられる。この場合、予測式は

$$y_P(t+j) \quad = \quad y_M(t+j) + y(t) - y_M(t) \tag{1.15}$$

で与えられる。この補正項の意味は、「現時刻でのプロセスとモデルとの差 $y(t) - y_M(t)$、すなわちモデル化誤差や外乱などの影響は将来一定に保たれる。」というものである。あるいは、モデルとプロセスとが一致している状況下でのステップ状外乱の影響とみることもできる。

1.4 参照軌道と制御則

初めに述べたように、設定値 r と予測値 y_P とが一致するように操作量を決定すればよいのであるから、制御則として、例えば

$$y_P(t+1) \quad = \quad r(t+1) \tag{1.16}$$

を用いればよいと思われるかもしれない。しかし現実には、モデル化誤差の影響のため、このような制御則が希望通りに機能することは極めて稀である。この制御則の最大の問題は、チューニングパラメータが存在しないことである。制御系に要求される仕様としてまず挙げられるのは制御性能であろう。しかし、制御性能と同程度にロバスト性も重要である。制御系のロバスト性とは、ある制御系が許容できるモデル化誤差、あるいはプロセスの特性変化の程度と解釈することができるが、これは制御性能とは相反する仕様である。すなわち、制御性能を良くしようとすればロバスト性が低下し、ロバスト性を良くしようとすれば制御性能が低下する。従って、制御性能とロバスト性との妥協点を探ることが重要であり、この妥協を可能にするために存在するのがチューニングパラメータである。

MPC にチューニングパラメータを持ち込む方法として広く利用されているのは、次の 2 つである。

- 参照軌道を導入する
- 操作量に対するペナルティーを導入する

ここでは参照軌道について述べることにしよう。操作量に対するペナルティーに関しては改めて詳述することにする。

制御則を Eq.(1.16) とすると、モデル化誤差が存在する場合に、操作量が必要以上に大きく変化し、結果として制御性能が悪くなることが考えられる。このような状況を回避するためには、徐々に設定値に近づくような目標値を作り、予測値がその目標値と一致するように操作量を決めるという方法が考えられる。このような目標値の描く軌道は**参照軌道** (Reference Trajectory) と呼ばれ、例えば次式で与えられる。

$$y_R(t+j) \quad = \quad \alpha^j y(t) + (1-\alpha^j) r(t+j) \qquad \text{for } j > 0 \tag{1.17}$$

ここで、α がチューニングパラメータであり、$\alpha = 0$ のとき参照軌道は設定値と一致し、$\alpha = 1$ のとき参照軌道は現時刻の出力 $y(t)$ で一定となる。すなわち、α を小さくすることによって制御系の速応性を高めるこ

とが可能であり、逆に α を大きくすることによって制御系のロバスト性を向上させることが可能である。なお、設定値が一定である場合には、参照軌道は $y(t)$ を始点とする 1 次遅れ系のステップ応答に等しくなる。

参照軌道を用いる場合、制御則は Eq.(1.17) の代わりに

$$y_P(t+1) \quad = \quad y_R(t+1) \tag{1.18}$$

となる。

1.5 一段予測制御アルゴリズム

前節の議論をふまえて、最も簡単な 1 段予測制御アルゴリズムを述べる。1 段予測制御という名称は、次のサンプリング時刻 (1 ステップ先) において予測値と参照軌道が一致するように操作量を決定することに由来する。

ここでは、プロセスモデルとしてインパルス応答モデルを用いることにする。

＜モデル＞

$$y_M(t) \quad = \quad dG_M(d)u(t) \tag{1.19}$$

予測式および参照軌道は次のようになる。

＜予測式＞

$$\begin{aligned} y_P(t+1) \quad &= \quad y_M(t+1) + y(t) - y_M(t) \\ &= \quad (1-d)G_M(d)u(t) + y(t) \end{aligned} \tag{1.20}$$

＜参照軌道＞

$$y_R(t+1) \quad = \quad \alpha y(t) + (1-\alpha)r(t+1) \tag{1.21}$$

制御則は Eq.(1.18) と同じく、

＜制御則＞

$$y_P(t+1) \quad = \quad y_R(t+1) \tag{1.22}$$

となるので、操作量 $u(t)$ について解くと、最終的に

$$u(t) \quad = \quad \frac{1-\alpha}{(1-d)G_M(d)}\{r(t+1) - y(t)\} \tag{1.23}$$

を得る。得られた制御系をブロック線図を用いて表現すると Fig.1.3 のようになる。コントローラの伝達関数は Eq.(1.23) によって規定されるが、コントローラの伝達関数の分母が $1-d$ を因数として持つことに注意すべきである。この事実はコントローラが積分要素を持つことを意味しており、結果として Fig.1.3 に示した制御系は 1 型となっている。すなわち、ステップ状の外乱および設定値変更に対して定常偏差が残らない制御系になっている。

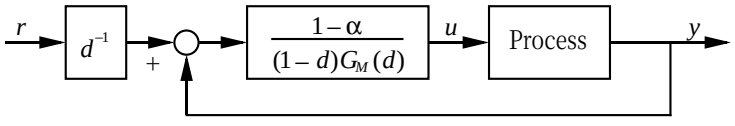

Fig. 1.3 1 段予測制御のブロック線図

コントローラが積分要素を持つに至った理由は、Eq.(1.20) から明らかであろう。すなわち、予測式で用いる未来の時刻と現時刻の推定値の差がコントローラの積分要素を生み出しているのである。逆に、1 型の制御系を実現するために、すなわちコントローラに積分要素を持たせるために、内部モデル原理に従って予測式を Eq.(1.20) のように定めたとも言える。

次に、別の角度から 1 段予測制御系をみてみよう。Eq.(1.23) の代わりにモデル $y_M(t)$ を陽に含んだ形で、操作量 $u(t)$ を

$$u(t) \quad = \quad \frac{1-\alpha}{(1-\alpha d)G_M(d)}\{r(t+1) - y(t) + y_M(t)\} \tag{1.24}$$

と表すこともできる。この場合の制御系は Fig.1.4 のようになるが、これは IMC(Internal Model Control) で用いられるブロック線図である。

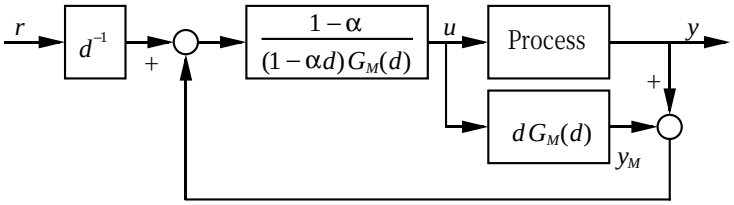

Fig. 1.4　1 段予測制御のブロック線図 (IMC 表現)

第2章　アルゴリズム

2.1　アルゴリズムの基本概念

Chap.1 では 1 段予測制御について述べたが、そのアルゴリズムは実際に利用されている MPC アルゴリズムに比べると非常に単純である。ここでは、第 1 世代に属する、より一般的な MPC アルゴリズムを詳述する。ちなみに、現在市販されている MPC は概ね第 3 世代に分類されるアルゴリズムを利用している。世代が進むにつれて、アルゴリズムはより高度化・複雑化するが、その詳細については割愛する。

一般的なアルゴリズムと言っても、基本概念は同じである。すなわち、プロセスの動的モデルに基づいて未来の被制御量の変化を予測し、被制御量と設定値 (参照軌道) ができるだけ近づくように操作量を決定する。ただし、次のサンプリング時刻のみならず、さらに先までも考慮する点が異なる。ここでは、Fig.2.1 を用いて MPC アルゴリズムの基本概念を説明する。

まず、現時刻 t において出力 $y(t)$ を測定し、$y(t)$ を始点として設定値 r に徐々に近づく参照軌道 $y_R(t+j)$ を計算する。次に、予測式を用いて未来の出力の予測値 $y_P(t+j)$ を求め、L ステップ先から P ステップに渡る区間 $[L, L+P-1]$ において予測値が参照軌道にできるだけ近づくように、現時刻以降 M ステップ間の入力 $u(t), u(t+1), \cdots, u(t+M-1)$ を決定する。得られた入力のうち現時刻に対応する $u(t)$ のみを実際にプロセスに加え、次のサンプリング時刻 $t+1$ までその値に保持する。

時刻 $t+1$ において出力 $y(t+1)$ が測定されれば、改めて時刻 $t+1$ を現時刻とみなし、未来の予測値が参照軌道とできるだけ近づくような入力を決定し、次のサンプリング時刻までその入力をプロセスに加える。以下、この手順を繰り返す。

このアルゴリズムにおいて、出力を予測する区間 $[1, L+P-1]$ を**予測区間** (Prediction Horizon)、予測値と参照軌道 (設定値) を一致させようとする区間 $[L, L+P-1]$ を**一致区間** (Coincidence Horizon)、入力の決定区間 M を**制御区間** (Control Horizon) と呼ぶ。

以下の数節では、この基本概念をアルゴリズムとして計算機上で実現させるために必要な項目について述べる。

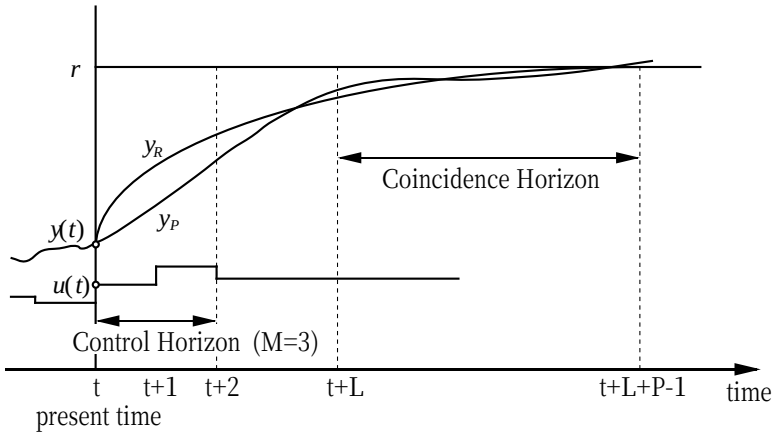

Fig. 2.1　MPC アルゴリズムの基本概念

2.2 プロセスモデル

予測にはモデルが必要であるが、インパルス応答モデルおよびステップ応答モデルについては Sec.1.2 で既に述べた。本節でもこれらのモデルを用いるが、一致区間 $[L, L+P-1]$ での予測値が必要となるため、行列による表現を利用する。

2.2.1 インパルス応答モデル

インパルス応答モデルは

$$y_M(t+j) \quad = \quad \sum_{k=1}^{s} h_k u(t+j-k) \tag{2.1}$$

で与えられる。現時刻を t として、過去の入力と現時刻以降の入力とを分けると、

$$y_M(t+j) \quad = \quad \sum_{k=1}^{j} h_k u(t+j-k) + \sum_{k=j+1}^{s} h_k u(t+j-k) \tag{2.2}$$

となる。さらに、現時刻 t から M ステップ間の入力が $u(t), u(t+1), \cdots, u(t+M-1)$ であり、時刻 $t+M$ 以降の入力は $u(t+M-1)$ に等しい、すなわち

$$u(t+i) \quad = \quad u(t+M-1) \qquad \text{for } i \geq M-1 \tag{2.3}$$

と仮定すると、

$$\begin{aligned} y_M(t+j) \quad &= \quad \sum_{k=1}^{j-M+1} h_k u(t+M-1) + h_{j-M+2} u(t+M-2) + \cdots + h_{j-1} u(t+1) + h_j u(t) \\ &+ \quad \sum_{k=1}^{s-j} h_{j+k} u(t-k) \end{aligned} \tag{2.4}$$

を得る。

MPC アルゴリズムでは一致区間 $[L, L+P-1]$ での予測値が必要となるため、この区間に対応するモデルをまとめて表現すると、

$$\begin{aligned} \begin{pmatrix} y_M(t+L) \\ y_M(t+L+1) \\ \vdots \\ y_M(t+L+P-1) \end{pmatrix} &= \begin{pmatrix} h_L & h_{L-1} & \cdots & h_{L-M+2} & \sum_{k=1}^{L-M+1} h_k \\ h_{L+1} & h_L & \cdots & h_{L-M+3} & \sum_{k=1}^{L-M+2} h_k \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ h_{L+P-1} & h_{L+P-2} & \cdots & h_{L+P-M+1} & \sum_{k=1}^{L+P-M} h_k \end{pmatrix} \begin{pmatrix} u(t) \\ u(t+1) \\ \vdots \\ u(t+M-1) \end{pmatrix} \\ &+ \begin{pmatrix} h_{L+1} & h_{L+2} & \cdots & h_{s-1} & h_s \\ h_{L+2} & h_{L+3} & \cdots & h_s & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ h_{L+P} & h_{L+P+1} & \cdots & 0 & 0 \end{pmatrix} \begin{pmatrix} u(t-1) \\ u(t-2) \\ \vdots \\ u(t-s+L) \end{pmatrix} \end{aligned} \tag{2.5}$$

となる。現時刻 t の推定値が

$$y_M(t) \quad = \quad \sum_{k=1}^{s} h_k u(t-k) \tag{2.6}$$

で与えられることを考慮すると、モデルはさらに

$$
\begin{pmatrix} y_M(t+L) \\ y_M(t+L+1) \\ \vdots \\ y_M(t+L+P-1) \end{pmatrix} = \begin{pmatrix} y_M(t) \\ y_M(t) \\ \vdots \\ y_M(t) \end{pmatrix}
$$

$$
+ \begin{pmatrix} h_L & h_{L-1} & \cdots & h_{L-M+2} & \sum_{k=1}^{L-M+1} h_k \\ h_{L+1} & h_L & \cdots & h_{L-M+3} & \sum_{k=1}^{L-M+2} h_k \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ h_{L+P-1} & h_{L+P-2} & \cdots & h_{L+P-M+1} & \sum_{k=1}^{L+P-M} h_k \end{pmatrix} \begin{pmatrix} u(t) \\ u(t+1) \\ \vdots \\ u(t+M-1) \end{pmatrix}
$$

$$
+ \begin{pmatrix} h_{L+1}-h_1 & h_{L+2}-h_2 & \cdots & h_s-h_{s-L} & \cdots & -h_s \\ h_{L+2}-h_1 & h_{L+3}-h_2 & \cdots & 0-h_{s-L} & \cdots & -h_s \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ h_{L+P}-h_1 & h_{L+P+1}-h_2 & \cdots & 0-h_{s-L} & \cdots & -h_s \end{pmatrix} \begin{pmatrix} u(t-1) \\ u(t-2) \\ \vdots \\ u(t-s) \end{pmatrix} \tag{2.7}
$$

と表現することができる。簡単のため、このモデルを次のように表すことにする。

$$
\boldsymbol{Y}_M = \boldsymbol{Y}_{M0} + \boldsymbol{H}_f \boldsymbol{u}_f + \boldsymbol{H}_o \boldsymbol{u}_o \tag{2.8}
$$

ここで、添え字 f, o はそれぞれ未来および過去を意味している。

2.2.2 ステップ応答モデル

ステップ応答モデルは

$$
y_M(t+j) = \sum_{k=1}^{\infty} a_k \Delta u(t+j-k) \tag{2.9}
$$

で与えられる。現時刻を t として、過去の入力と現時刻以降の入力とを分けると、

$$
\begin{aligned} y_M(t+j) &= \sum_{k=1}^{j} a_k \Delta u(t+j-k) + \sum_{k=j+1}^{\infty} a_k \Delta u(t+j-k) \\ &= \sum_{k=0}^{j-1} a_{j-k} \Delta u(t+k) + \sum_{k=1}^{\infty} a_{j+k} \Delta u(t-k) \end{aligned} \tag{2.10}
$$

となる。さらに、時刻 $t+M$ 以降の入力は $u(t+M-1)$ に等しい、すなわち

$$
\Delta u(t+i) = 0 \qquad \text{for } i \geq M \tag{2.11}
$$

と仮定すると、

$$
y_M(t+j) = \sum_{k=0}^{M-1} a_{j-k} \Delta u(t+k) + \sum_{k=1}^{\infty} a_{j+k} \Delta u(t-k) \tag{2.12}
$$

を得る。

ここまでの手順はインパルス応答モデルと同じである。しかし、ステップ応答モデルの場合、$y_M(t+j)$ を計算するために無限個の入力データが必要であるため、このままでは現実的なモデルといえない。そこで、現時刻 t の推定値が

$$y_M(t) \quad = \quad \sum_{k=1}^{\infty} a_k \Delta u(t-k) \tag{2.13}$$

で与えられることを考慮して、Eq.(2.12) で与えられる推定値 $y_M(t+j)$ を現時刻の推定値 $y_M(t)$ からの変化として表現すると、

$$y_M(t+j) \quad = \quad y_M(t) + \sum_{k=0}^{M-1} a_{j-k} \Delta u(t+k) + \sum_{k=1}^{s-1} (a_{j+k} - a_k) \Delta u(t-k) \tag{2.14}$$

を得る。これにより、有限個の入力データから推定値 $y_M(t+j)$ を計算することが可能になる。

MPC アルゴリズムでは一致区間 $[L, L+P-1]$ での予測値が必要となるため、この区間に対応するモデルをまとめて表現すると、

$$\begin{aligned}
\begin{pmatrix} y_M(t+L) \\ y_M(t+L+1) \\ \vdots \\ y_M(t+L+P-1) \end{pmatrix} &= \begin{pmatrix} y_M(t) \\ y_M(t) \\ \vdots \\ y_M(t) \end{pmatrix} \\
&+ \begin{pmatrix} a_L & a_{L-1} & \cdots & a_{L-M+1} \\ a_{L+1} & a_L & \cdots & a_{L-M+2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{L+P-1} & a_{L+P-2} & \cdots & a_{L+P-M} \end{pmatrix} \begin{pmatrix} \Delta u(t) \\ \Delta u(t+1) \\ \vdots \\ \Delta u(t+M-1) \end{pmatrix} \\
&+ \begin{pmatrix} a_{L+1} - a_1 & a_{L+2} - a_2 & \cdots & a_{L+s-1} - a_{s-1} \\ a_{L+2} - a_1 & a_{L+3} - a_2 & \cdots & a_{L+s} - a_{s-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{L+P} - a_1 & a_{L+P+1} - a_2 & \cdots & a_{L+s+P-1} - a_{s-1} \end{pmatrix} \begin{pmatrix} \Delta u(t-1) \\ \Delta u(t-2) \\ \vdots \\ \Delta u(t-s+1) \end{pmatrix}
\end{aligned} \tag{2.15}$$

と表現することができる。簡単のため、このモデルを次のように表すことにする。

$$\boldsymbol{Y}_M \quad = \quad \boldsymbol{Y}_{M0} + \boldsymbol{A}_f \Delta \boldsymbol{u}_f + \boldsymbol{A}_o \Delta \boldsymbol{u}_o \tag{2.16}$$

2.3 予測式と参照軌道

モデル化誤差や外乱の影響を考慮し、Sec.1.3 で述べた次の予測式

$$y_P(t+j) \quad = \quad y_M(t+j) + y(t) - y_M(t) \tag{2.17}$$

を用いるものとする。予測式についても、一致区間 $[L, L+P-1]$ に対応する式をまとめて表現すると、

$$\boldsymbol{Y}_P \quad = \quad \boldsymbol{Y}_M + \boldsymbol{Y} - \boldsymbol{Y}_{M0} \tag{2.18}$$

となる。ここで、

$$\boldsymbol{Y}_P \quad = \quad \begin{pmatrix} y_P(t+L) \\ y_P(t+L+1) \\ \vdots \\ y_P(t+L+P-1) \end{pmatrix} \tag{2.19}$$

$$\boldsymbol{Y} = \begin{pmatrix} y(t) \\ y(t) \\ \vdots \\ y(t) \end{pmatrix} \tag{2.20}$$

である。

MPC アルゴリズムは、制御系のロバスト性を確保するために参照軌道を用いるものと操作量に対するペナルティーを評価に取り入れるものとに大別されるが、ここでは Sec.1.4 で述べた参照軌道を用いることにする。ただし、一致区間が L ステップ先から始まることを考慮して、参照軌道を次式で与える。

$$y_R(t+j) = \alpha^{j-L+1} y(t) + (1-\alpha^{j-L+1}) r(t+j) \quad \text{for } j \geq L \tag{2.21}$$

予測式と同様、一致区間 $[L, L+P-1]$ に対応する参照軌道をまとめて表現すると、

$$\boldsymbol{Y}_R = \boldsymbol{\alpha}\boldsymbol{Y} + (\boldsymbol{I} - \boldsymbol{\alpha})\boldsymbol{r} \tag{2.22}$$

となる。ここで、

$$\boldsymbol{Y}_R = \begin{pmatrix} y_R(t+L) \\ y_R(t+L+1) \\ \vdots \\ y_R(t+L+P-1) \end{pmatrix} \tag{2.23}$$

$$\boldsymbol{r} = \begin{pmatrix} r(t+L) \\ r(t+L+1) \\ \vdots \\ r(t+L+P-1) \end{pmatrix} \tag{2.24}$$

$$\boldsymbol{\alpha} = \begin{pmatrix} \alpha & & & 0 \\ & \alpha^2 & & \\ & & \ddots & \\ 0 & & & \alpha^P \end{pmatrix} \tag{2.25}$$

である。

2.4 制御則

1 段予測制御 (Sec.1.5) においては、予測値が参照軌道 (目標値) に一致するように操作量を決定した。しかし、数ステップに及ぶ一致区間 ($P > 1$) を持つアルゴリズムを利用する場合、予測値が正確に参照軌道と一致することは望めない。そこで、予測値と参照軌道の近さを表す指標が必要となるが、通常、ユークリッドノルム (の二乗) が用いられる。この場合、MPC の制御則は、

$$J = \|\boldsymbol{Y}_P - \boldsymbol{Y}_R\|^2 \tag{2.26}$$

で定義される評価関数 J を最小にする操作量 $\{u(t+j)\}(j = 0, 1, \cdots, M-1)$ を求める問題として定式化される。

まず、インパルス応答モデルを用いる場合について考えてみよう。この場合、予測式は Eqs.(2.8),(2.18) より、

$$\boldsymbol{Y}_P = \boldsymbol{Y} + \boldsymbol{H}_f \boldsymbol{u}_f + \boldsymbol{H}_o \boldsymbol{u}_o \tag{2.27}$$

で与えられるので、評価関数 J は

$$J = (\boldsymbol{Y} + \boldsymbol{H}_f\boldsymbol{u}_f + \boldsymbol{H}_o\boldsymbol{u}_o - \boldsymbol{Y}_R)^T(\boldsymbol{Y} + \boldsymbol{H}_f\boldsymbol{u}_f + \boldsymbol{H}_o\boldsymbol{u}_o - \boldsymbol{Y}_R) \tag{2.28}$$

となる。現時刻以降の操作量 $\boldsymbol{u}_f$ を変化させた場合に、評価関数 J が最小となるための必要条件は

$$\frac{\partial J}{\partial \boldsymbol{u}_f} = \boldsymbol{0} \tag{2.29}$$

である。この条件式を実際に計算すると、

$$\begin{aligned}\frac{\partial J}{\partial \boldsymbol{u}_f} &= \begin{pmatrix} \dfrac{\partial J}{\partial u(t)} \\ \dfrac{\partial J}{\partial u(t+1)} \\ \vdots \\ \dfrac{\partial J}{\partial u(t+M-1)} \end{pmatrix} \\ &= 2\boldsymbol{H}_f^T\boldsymbol{H}_f\boldsymbol{u}_f + 2\boldsymbol{H}_f^T(\boldsymbol{Y} + \boldsymbol{H}_o\boldsymbol{u}_o - \boldsymbol{Y}_R) \\ &= \boldsymbol{0}\end{aligned} \tag{2.30}$$

となるので、$\boldsymbol{H}_f^T\boldsymbol{H}_f$ が正則である場合には、

$$\boldsymbol{u}_f = (\boldsymbol{H}_f^T\boldsymbol{H}_f)^{-1}\boldsymbol{H}_f^T(\boldsymbol{Y}_R - \boldsymbol{Y} - \boldsymbol{H}_o\boldsymbol{u}_o) \tag{2.31}$$

を得る。

ステップ応答モデルを用いる場合についても同様である。すなわち、予測式が Eqs.(2.16),(2.18) を用いて

$$\boldsymbol{Y}_P = \boldsymbol{Y} + \boldsymbol{A}_f\Delta\boldsymbol{u}_f + \boldsymbol{A}_o\Delta\boldsymbol{u}_o \tag{2.32}$$

と与えられるので、現時刻以降の操作量 $\Delta\boldsymbol{u}_f$ は、$\boldsymbol{A}_f^T\boldsymbol{A}_f$ が正則である場合には、

$$\Delta\boldsymbol{u}_f = (\boldsymbol{A}_f^T\boldsymbol{A}_f)^{-1}\boldsymbol{A}_f^T(\boldsymbol{Y}_R - \boldsymbol{Y} - \boldsymbol{A}_o\Delta\boldsymbol{u}_o) \tag{2.33}$$

によって決定される。

2.5 操作量に対するペナルティーを考慮した制御則

制御系のロバスト性を確保するために、操作量に対するペナルティーを評価関数に組み込む方法がある。ここでは、その方法を採用した場合の制御則について述べる。

MPC の最も原始的な考え方に従えば、予測値と設定値とが一致するように操作量を決定すればよいことになる。しかし、モデル化誤差が無視できない場合には、このような単純な制御則によって高い制御性能を実現することはできないだけでなく、制御系が不安定になることも考えられる。この原因として操作量の急激な変化が挙げられるので、制御系のロバスト性を確保するために、被制御量を瞬時に設定値に移すのではなく、徐々に設定値に近づけていくという方法が考えられる。この方法を具体化したものが参照軌道である。

一方、操作量の急激な変化を避ければよいという考え方をそのまま制御則の一部として利用する方法も考えられる。この場合の評価関数 J は例えば次式のようになる。

$$J = \|\boldsymbol{Y}_P - \boldsymbol{Y}_R\|_Q^2 + \|\Delta\boldsymbol{u}_f\|_R^2 \tag{2.34}$$

$$\|\boldsymbol{Y}_P - \boldsymbol{Y}_R\|_Q^2 = (\boldsymbol{Y}_P - \boldsymbol{Y}_R)^T\boldsymbol{Q}(\boldsymbol{Y}_P - \boldsymbol{Y}_R) \tag{2.35}$$

$$\|\Delta\boldsymbol{u}_f\|_R^2 = \Delta\boldsymbol{u}_f^T\boldsymbol{R}\Delta\boldsymbol{u}_f \tag{2.36}$$

ここで、$\boldsymbol{Q}, \boldsymbol{R}$ は共に非負定値行列であり、通常は対角行列とする。これらの重み行列の各要素がチューニングパラメータとなる。なお、通常は、参照軌道 $\boldsymbol{Y}_R$ の代わりに設定値 $\boldsymbol{r}$ がそのまま用いられる。この評価関数の第 1 項は予測値と参照軌道 (設定値) とを近づける効果を持ち、第 2 項は操作量の急激な変動を抑制する効果を持つ。従って、重み行列 $\boldsymbol{R}$ を大きくすれば、制御系の速応性を犠牲にしてロバスト性を向上させることができる。逆に、重み行列 $\boldsymbol{R}$ を小さくすれば、制御系のロバスト性を犠牲にして速応性を向上させることができる。

それでは、Eq.(2.34) の評価関数を最小にする操作量を求めてみよう。ここではステップ応答モデルを用いることにする。このとき、予測式は Eq.(2.32) で与えられるので、現時刻以降の操作量 $\Delta\boldsymbol{u}_f$ は、$\boldsymbol{A}_f^T\boldsymbol{Q}\boldsymbol{A}_f+\boldsymbol{R}$ が正則である場合には

$$\Delta\boldsymbol{u}_f \quad = \quad (\boldsymbol{A}_f^T\boldsymbol{Q}\boldsymbol{A}_f + \boldsymbol{R})^{-1}\boldsymbol{A}_f^T\boldsymbol{Q}(\boldsymbol{Y}_R - \boldsymbol{Y} - \boldsymbol{A}_o\Delta\boldsymbol{u}_o) \tag{2.37}$$

と求めることができる。

操作量に対するペナルティーを考慮した評価関数は一般に

$$J \quad = \quad \|\boldsymbol{Y}_P - \boldsymbol{Y}_R\|_Q^2 + \|\Delta\boldsymbol{u}_f\|_R^2 + \|\boldsymbol{u}_f - \boldsymbol{u}_{ss}\|_S^2 \tag{2.38}$$

と表現することができる。第 1,2 項が予測値と目標値とを一致させる効果および操作量の急激な変動を抑制する効果を持つことは上述の通りである。第 3 項は操作量を希望する定常値 $\boldsymbol{u}_{ss}$ に近づける効果を持つ。通常、定常値 $\boldsymbol{u}_{ss}$ は MPC の上位に位置するオプティマイザーから与えられる。なお、重み行列 $\boldsymbol{S}$ は $\boldsymbol{Q}$,$\boldsymbol{R}$ と同様非負定値行列であるが、被制御量を設定値に一致させることが重要であるため、その (i,j) 要素を

$$s_{ij}\begin{cases} \neq 0 & \text{for } i=j=M-1 \\ = 0 & \text{for other } i,j \end{cases} \tag{2.39}$$

とすることが多い。

操作量の急激な変化を抑制する方法を 2 種類紹介したが、参照軌道を用いるべきか、操作量に対するペナルティーを用いるべきかを判断するための明確な基準はない。従って、MPC の適用対象と制御目的を考慮して、目標の実現が容易であると思われる方法を採用するほかない。

2.6 モデルの構築

MPC の性能は利用するプロセスモデルの精度に強く依存するため、モデルの構築は慎重に行われなければならない。ここでは、インパルス応答モデルあるいはステップ応答モデルを利用する場合を想定し、

- パラメータの決定
- 打ち切り次数の決定

という観点からモデル構築方法について述べる。

統計的モデルを構築するためには、同定実験が必要である。すなわち、同定のためのテスト信号をプロセスに加え、結果として得られる入出力データからモデルを構築しなければならない。いま、ステップ状入力をプロセスに付加した場合を考えよう。この際得られる被制御量の応答波形はステップ応答に他ならない。では、この応答波形から直接ステップ応答係数を読みとればよいのであろうか。当然ダメである。なぜなら、同定実験から得られるデータは外乱や測定ノイズの影響を受けて汚されているからである。外乱や測定ノイズの影響を受けにくいモデルを構築するために、まず低次の伝達関数モデルなどのパラメトリックモデルを同定し、得られたモデルをインパルス応答モデルあるいはステップ応答モデルに変換するという方法が一般的に利用されている。このように一度低次のモデルを同定することによって、**過剰パラメータ表現** (over-parameterization) を避けることができる。過剰なパラメータを用いて同定を行うと、同定に用い

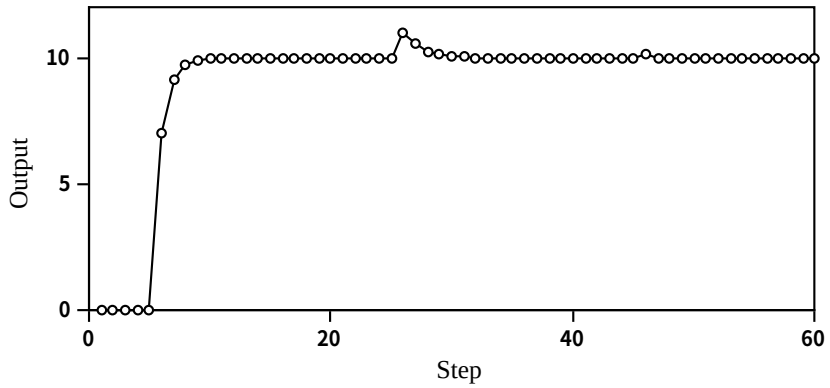

Fig. 2.2　打ち切り次数の影響

たデータそのものにはよく適合するが、その他のデータには適合しないモデルが得られる。あるいは、様々な運転状況に適合するモデルを構築するためには、パラメータの数は少ない方がよいと表現してもよい。これはいわゆる「けちの原理」として知られている事実である。

これと同じ状況は、テスト信号としてステップ以外の入力を付加した場合についても生じる。従って、モデルを構築する際に、インパルス応答係数あるいはステップ応答係数をパラメータとして直接フィッティングしてはいけないことに注意すべきである。同定用のパラメトリックモデルとしては、例えば伝達関数モデルであれば 2 次遅れ＋むだ時間程度で十分であることが多い。

インパルス応答モデルあるいはステップ応答モデルを利用する際のもう 1 つの問題は、打ち切り次数の決定である。Sec.1.2 で述べたように、応答波形の変化が無視できるほど小さくなる次数 s までの応答係数をモデルとして採用するわけであるが、打ち切り次数 s を大きくしすぎると計算機内に保存すべきパラメータ数が膨大になり、計算負荷も増大する。逆に小さくしすぎると、応答が途中で打ち切られてしまうため、MPC を適用する際に、打ち切り次数と等しいステップごとに制御応答が乱れる現象が発生する。例として、プロセスとモデルの伝達関数が共に

$$G(s) = \frac{2}{10s+1} \tag{2.40}$$

である場合を想定し、サンプリング間隔を 1、打ち切り次数を 20 としてモデル予測制御を適用した場合のシミュレーション結果を Fig.2.2 に示す。シミュレーション開始後 5 ステップで設定値を変更しているが、設定値変更から 20 ステップの時点で制御応答が乱れていることが確認できる。その後 20 ステップごとに応答が乱れるが、その影響は徐々に小さくなっている。従って、このような制御応答の乱れが無視できる程度に打ち切り次数を大きくする必要がある。

打ち切り次数 s が大きいときに計算負荷を低減するために、インパルス応答モデルあるいはステップ応答モデルと ARX モデルとを併用することもできる。例えば、ステップ応答モデルを用いる場合について考えてみよう。このとき、操作量は

$$\Delta \boldsymbol{u}_f \quad = \quad (\boldsymbol{A}_f^T \boldsymbol{A}_f)^{-1} \boldsymbol{A}_f^T (\boldsymbol{Y}_R - \boldsymbol{Y} - \boldsymbol{A}_o \Delta \boldsymbol{u}_o) \tag{2.41}$$

で与えられるが、行列 $\boldsymbol{A}_f$ のサイズはパラメータ L, P, M で規定されており、打ち切り次数 s には依存しない。一方、$\boldsymbol{Y} + \boldsymbol{A}_o \Delta \boldsymbol{u}_o$ は未来の操作量を $u(t-1)$ で一定にする場合の予測値であるから、この予測値を ARX モデルに基づいて算出することにより、打ち切り次数 s の大きさに起因する計算負荷の問題は解消できる。

2.7　制御パラメータの調節

ここまでで述べたMPCアルゴリズムには、$L, M, P, \alpha, \boldsymbol{Q}, \boldsymbol{R}, \boldsymbol{S}$という制御パラメータが用いられている。これらすべてのパラメータを適切に定める作業が大変困難であることは容易に想像できる。このため、いくつかのパラメータは予め固定しておき、少数のパラメータをオンラインチューニングに用いるという方法が一般的である。この際、予め固定するパラメータの値が問題となるが、その決定に関して以下の経験則が広く利用されている。なお、この経験則では、$L = 1, \boldsymbol{S} = \boldsymbol{0}$としている。

1. PT_sを開ループの時定数と同程度に設定する。ここで、T_sは制御周期である。通常は$P > 20$とする。
2. Mを3〜5に設定する。
3. ＜参照軌道を用いる場合＞
   αをオンラインチューニングに用いる。制御が強い場合にはαを1に近づけ、弱い場合には0に近づける。
   ＜入力に対するペナルティーを用いる場合＞
   (a) $\boldsymbol{Q}$は一定値を対角要素に持つ行列とする。ここでは、SISO系を考えているので、$\boldsymbol{Q}$は単位行列でよい。
   (b) $\boldsymbol{R}$をオンラインチューニングに用いる。制御が強い場合には$\boldsymbol{R}$を大きくし、弱い場合には小さくする。この際、$\boldsymbol{R}$の対角要素はすべて同じ値とする。
4. 制御応答が極めて遅い場合には、$\alpha = 0$あるいは$\boldsymbol{R} = \boldsymbol{0}$とした上で、適当に$P$を小さく$M$を大きくする。