2.4 广义预测控制（GPC)

广义预测控制是 Clarke 等 1987 年提出的基于参数模型的控制算法，它是以受控自回归积分滑动平均（CARIMA) 模型为基础，并结合了辨识和自校正机制，具有良好的鲁棒性。

丢番图方程（Diophantine Equation） 有一个或者几个变量的整系数方程，它们的求解仅仅在整数范围内进行。最后这个限制使得丢番图方程求解与实数范围方程求解有根本的不同。丢番图方程又名不定方程、整系数多项式方程，是变量仅容许是整数的多项式等式。

CARIMA model If a system is given by the discrete form

$$\alpha_0 y(k)+\alpha_1 y(k-1)+\cdots+\alpha_N y(k-N)=\beta_1 u(k-1)+\beta_2 u(k-2)+\cdots+\beta_N u(k-N)+\varepsilon(k), \quad (2.3)$$

where $y(\cdot)$ are the outputs, $u(\cdot)$ are the inputs, $\xi(\cdot)$ are the noises, with α_i, β_i, $i=0,1,\ldots,N$ being the weights. Introducing the time delay factor z^{-1}, the above equation can be reformatted into

$$\alpha_0 y(k)+\alpha_1 z^{-1} y(k)+\cdots+\alpha_N z^{-N} y(k)=\beta_1 u(k-1)+\beta_2 z^{-1} u(k-1)+\cdots+\beta_N z^{N-1} u(k-1)+\varepsilon(k).$$

In order to introduce the differentiating operator $\Delta=1-z^{-1}$[2], multiply both sides of the above equation with z^{-1} yielding

$$\begin{aligned}&z^{-1}\left[\alpha_0 y(k)+\alpha_1 z^{-1} y(k)+\cdots+\alpha_N z^{-N} y(k)\right]\\ =&z^{-1}\left[\beta_1 u(k-1)+\beta_2 z^{-1} u(k-1)+\cdots+\beta_N z^{N-1} u(k-1)+\varepsilon(k)\right].\end{aligned}$$

Substraction of the above two equations gives

$$\begin{aligned}&\alpha_0\Delta y(k)+\alpha_1 z^{-1}\Delta y(k)+\cdots+\alpha_N z^{-N}\Delta y(k)\\ =&\beta_1\Delta u(k-1)+\beta_2 z^{-1}\Delta u(k-1)+\cdots+\beta_N z^{N-1}\Delta u(k-1)+\Delta\varepsilon(k),\end{aligned}$$

which also denotes as

$$A\left(z^{-1}\right)\Delta y(k)=B\left(z^{-1}\right)\Delta u(k-1)+\xi(k),$$

with

$$\begin{cases}A(z^{-1}) &= \alpha_0+\alpha_1 z^{-1}+\cdots+\alpha_N z^{-N},\\ B(z^{-1}) &= \beta_1+\beta_2 z^{-1}+\cdots+\beta_N z^{N-1},\\ \xi(k) &= \Delta\varepsilon(k).\end{cases}$$

[2] Actually, I don't know why.

NOTE　我们所学习的 GPC 将基于 CARIMA，但事实上，如果把 CARIMA 模型换成式 (2.3)，在推导和理解时将会更简单。而两者的区别仅在于，使用 CARIMA 模型时，在优化目标中可以使用 Δu，而常规离散模型中使用的是 u，因此在求解得到的结果分别是：使控制量尽量平稳/使控制量尽可能小。在直观感觉上，使控制量尽可能小的效果应该会更好一些，除非后文提及的 $\left(G^TG+\lambda I\right)^{-1}G^T$ 是奇异的，但是是显然不是。谜之操作。

2.4.1　预测模型

假设系统是基于下面的 CARIMA 模型[3]:

$$A\left(z^{-1}\right)y(k)=B\left(z^{-1}\right)u(k-1)+\xi(k)/\Delta \tag{2.4}$$

其中，$y(k)$，$u(k)$，$\xi(k)$ 是系统的输出、输入和干扰信号。

$$A\left(z^{-1}\right)=1+a_1z^{-1}+\ldots+a_{na}z^{-n_a}$$

$$B\left(z^{-1}\right)=b_0+b_1z^{-1}+\ldots+b_{n_b}z^{-n_b}$$

$$\Delta=1-z^{-1}$$

z^{-1} 是向后移时间算子，是 z 变换的逆算子。

Objective　We are already given the model (2.4), which can be transformed into

$$A\left(z^{-1}\right)\Delta y(k)=B\left(z^{-1}\right)\Delta u(k-1)+\xi(k), \tag{2.5}$$

and all we want is to predict the output at time k, e.g. $y\left(k\right)$. However, it's not a good idea to divide both sides with $A\left(z^{-1}\right)$ since $A\left(z^{-1}\right)$ is a polynomial. If we can multiply both sides with E_j, and then with some addends $z^{-j}F_j$, which[4] satisfy[5]

$$1=E_j\left(z^{-1}\right)A\left(z^{-1}\right)\Delta+z^{-j}F_j\left(z^{-1}\right),j=1\ldots,N,$$

[3]更详细的内容请参见附录。

[4]This equation is one example of Diophantine equation.

[5]Additional assumptions are given as $\partial E_j=j-1$，$\partial F_j=n_a$,

$$E_j=e_0+e_1z^{-1}+\ldots+e_{j-1}z^{-(j-1)},$$
$$F_j=f_{j0}+f_{j1}z^{-1}+\ldots+f_{jn_a}z^{-n_a}.$$

so as to solve $E_j\left(z^{-1}\right)$ and $F_j\left(z^{-1}\right)$.

so multiplying both sides of eq (2.5) with $E_j\left(z^{-1}\right)$, and then adding $z^{-j}F_j\left(z^{-1}\right)$ to both sides of the equation yields[6]

$$\begin{aligned} E_jA\Delta y(k)+z^{-j}F_jy(k) &= E_j\left[B\Delta u(k-1)+\xi(k)\right]+z^{-j}F_jy(k), \\ \Rightarrow \left[E_jA\Delta+z^{-j}F_j\right]y(k) &= E_jB\Delta u\left(k-1\right)+E\xi\left(k\right)+z^{-j}F_jy\left(k\right), \\ \Rightarrow y(k) &= E_jB\Delta u\left(k-1\right)+E\xi\left(k\right)+z^{-j}F_jy\left(k\right). \end{aligned}$$

In order to get the predictive at time $k+j$, multiply both sides with z^j, yielding

$$y(k+j)=E_jB\Delta u(k+j-1)+F_jy(k)+E_j\xi(k+j),$$

which means the predictive is

$$\hat{y}(k+j|k)=E_jB\Delta u\left(k+j-1\right)+F_jy(k). \tag{2.6}$$

Split the known/to-be-designe value of Δu　The eq (2.6) is used to predict the output under the control $\Delta u\left(\cdot\right)$, however, the $\Delta u\left(i\right)$, $i<k$ is previous assigned the driving mechanism and can not been re-assigned, so we should split the already-chosen values $\Delta u\left(i\right)$, $i<k$ and the to-be-design values $\Delta u\left(i\right),k\leq i\leq k+j$.

If there exist $G_j\left(z^{-1}\right)$, $H_j\left(z^{-1}\right)$ that satisfies[7]

$$E_jB=G_j+z^{-j}H_j,\quad j=1,\ldots,N,$$

together with this Diophantine equation, eq (2.6) can be transformed into

$$\hat{y}(k+j|k)=G_j\Delta u(k+j-1)+H_j\Delta u(k-1)+F_jy(k),$$

where $\Delta u\left(k-1\right)$ is complete known, and $\Delta u\left(k+j-1\right)$ is completely designable.

The reason to introduce Diophantine equations is dipicted in Figure 2.6 .

Matrix Notation　令 $y_0(k+j)=H_j\Delta u(k-1)+F_jy(k)$,

$$\hat{y}(k+j|k)=G_j\Delta u(k+j-1)+y_0(k+j). \tag{2.7}$$

[6]For brevity, all the z^{-1} in functions will be igored in display math, such that $A\left(z^{-1}\right)$ is denoted as A in display math.

[7]Additional assumptions are given as $\partial G_j=j-1,\partial H_j=n_b-1$,

$$\begin{aligned} G_j &= g_0+g_1z^{-1}+\ldots+g_{j-1}z^{-(j-1)}, \\ H_j &= h_{j0}+h_{j1}z^{-1}+\ldots+h_{jn_b-1}z^{-(n_b-1)}, \end{aligned}$$

so as to split the control inputs into two parts.

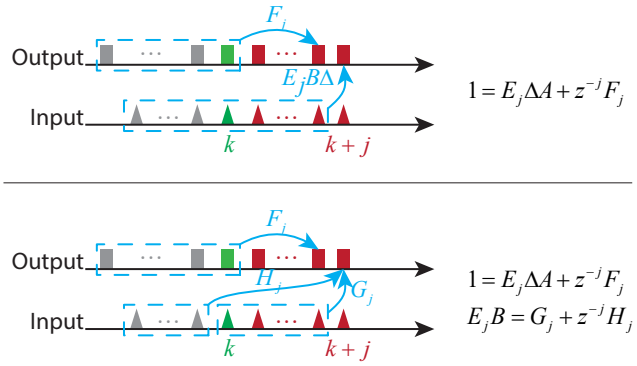

Figure 2.6: Effects of Diophantine Equations

令

$$\hat{Y}(k)=\begin{bmatrix}\hat{y}(k+1|k)\\ \hat{y}(k+2|k)\\ \cdots\\ \hat{y}(k+N|k)\end{bmatrix},\Delta U^*(k)=\begin{bmatrix}\Delta u(k)\\ \Delta u(k+1)\\ \cdots\\ \Delta u(k+N+1)\end{bmatrix},Y_0(k)=\begin{bmatrix}y_0(k+1)\\ y_0(k+2)\\ \cdots\\ y_0(k+N)\end{bmatrix},$$

于是式 (2.7) 可以写为向量形式：

$$\hat{Y}(k)=Y_0(k)+G^*\Delta U^*(k), \tag{2.8}$$

其中

$$G^*=\begin{bmatrix}g_0 & & & & 0\\ g_1 & g_0 & & & \\ g_2 & g_1 & g_0 & & \\ \vdots & \vdots & \vdots & \ddots & \\ g_{N-1} & g_{N-2} & g_{N-3} & \cdots & g_0\end{bmatrix}_{N\times N}.$$

引入控制步长 $N_u\le N$，并假设 $j>N_u$ 时，$\Delta u(k+j-1)=0$，式 (2.8) 可以改写成

$$\hat{Y}(k)=Y_0(k)+G\Delta U(k),$$

其中

$$G=\begin{bmatrix}g_0 & & 0\\ g_1 & \ddots & \\ \vdots & \ddots & g_0\\ \vdots & & g_1\\ g_{N-1} & \cdots & g_{N-N_u}\end{bmatrix}_{N^*N_u},\Delta U(k)=\begin{bmatrix}\Delta u(k)\\ \Delta u(k+1)\\ \cdots\\ \Delta u\left(k+N_u+1\right)\end{bmatrix}.$$

2.4.1.1　丢番图方程求解

利用递推算法求解丢番图方程:

1. $E_1 = 1, F_1 = z[1 - \overline{A}], \overline{A} = A\Delta,$
2. $E_{j+1} = E_j + e_j z^{-j}, F_{j+1} = z\left[F_j - e_j\overline{A}\right], j = 1, \ldots, N$
3. $e_j = f_0,$
4. $E_j B = G_j + z^{-j}H_j, \partial G_j = j - 1,$

Iteration Algorithm for H_j and G_j

$$\begin{aligned} E_j B &= G_j + z^{-j}H_j, \\ E_{j+1}B &= G_{j+1} + z^{-j-1}H_j, \end{aligned}$$

and $E_{j+1} = E_j + e_j z^{-j}$, $G_{j+1} = G_j + g_j z^{-j}$. And it's easy to deduct that

$$\begin{aligned} G_{j+1} + z^{-j-1}H_j &= E_j B + e_j z^{-j} B \\ &= G_j + z^{-j}H_j, \\ \Rightarrow g_j &= e_j B + H_j - z^{-1}H_{j+1}. \end{aligned}$$

Equaling the terms with/without z yields

$$\begin{aligned} g_{j+1} &= e_j b_0 + h_{j0}, \\ H_{j+1} &= z\left[(H_j - h_{j0}) + e_j\left(b_0 - B\right)\right]. \end{aligned}$$

2.4.2　跟踪轨迹

$$\begin{cases} y_d(k) = y(k), \\ y_d(k+j) = \alpha y_d(k+j-1) + (1-\alpha)y_r(k), j = 1, \ldots, N \end{cases} \tag{2.9}$$

其中 $y_r(k)$ 是当前设定值，$0 \le \alpha \le 1$ 为柔化因子。令

$$Y_d(k) = \begin{bmatrix} y_d(k+1) \\ y_d(k+2) \\ \vdots \\ y_d(k+N) \end{bmatrix}.$$

2.4.3　滚动优化

目标函数：

$$J=\sum_{j=1}^{N}\left[\hat{y}(k+j|k)-y_d(k+j)\right]^2+\lambda\sum_{j=1}^{N_n}[\Delta u(k+j-1)]^2,$$

其中 $\lambda>0$ 为控制加权因子。其对应的向量形式为

$$J=\left[\hat{Y}(k)-Y_d(k)\right]^T\left[\hat{Y}(k)-Y_d(k)\right]+\lambda[\Delta U(k)]^T[\Delta U(k)],$$

令 $\frac{\partial J}{\partial \Delta u(k)}=0$，得到控制律

$$\Delta U(k)=\left(G^TG+\lambda I\right)^{-1}G^T\left[Y_d(k)-Y_0(k)\right], \tag{2.10}$$

取 $\Delta U(k)$ 的第一个分量 $\Delta u(k)$ 为 k 时刻的控制输入量的改变量。于是，控制输入为

$$u(k)=u(k-1)+\Delta u(k). \tag{2.11}$$

2.4.4　反馈校正

GPC 的反馈校正主要是通过在线辨识模型系统构成的自校正控制算法实现。

假设系统模型 A，B 未知，设系统的模型为:

$$\hat{A}\left(z^{-1}\right)y(k)=\hat{B}\left(z^{-1}\right)u(k-1)+\xi(k)/\Delta,$$

$$\Delta y(k)=\left[1-\hat{A}\left(z^{-1}\right)\right]\Delta y(k)+\hat{B}\left(z^{-1}\right)\Delta u(k-1)=\hat{\theta}^T(k)\phi(k),$$

其中

$$\hat{\theta}(k)=\left[-\hat{a}_1,\ldots,-\hat{a}_{n_a},\hat{b}_0,\ldots,\hat{b}_{n_b}\right]^T,$$

$$\phi(k)=[\Delta y(k-1),\ldots,\Delta y(k-n_a),\Delta u(k-1),\ldots,\Delta u(k-n_b)]^T,$$

则：

$$\hat{\theta}(k)=\hat{\theta}(k-1)+\frac{P(k-1)\phi(k)}{1+\phi^T(k)P(k-1)\phi(k)}\left[\Delta y(k)-\hat{\theta}^T(k-1)\phi(k)\right],$$

$$P(k)=P(k-1)-\frac{P(k-1)\phi(k)\phi^T(k)P(k-1)}{1+\phi^T(k)P(k-1)\phi(k)}.$$

2.4.5　GPC 自校正算法流程

给定一些初值:θ_0，β，N，N_u，λ，α

1. k 时刻到来，采样获取 $y(k)$;
2. 在线辨识 $\hat{A}(z^{-1})$，$\hat{B}(z^{-1})$ 一次;
3. 用 $\hat{A}(z^{-1}), \hat{B}(z^{-1})$ 代替 $A(z^{-1}), B(z^{-1})$, 计算 E_j, F_j, G_j, H_j, 进而计算 $Y_0(k), G, \left(G^TG+\lambda I\right)^{-1}$;
4. 计算 $Y_d(k)$;
5. 求解控制量 $u(k)$;

数据回归，返回1。如果系统模型 A，B 已知，则2省略，3只需计算一次。

2.4.6　GPC 的内模结构

可由式 (2.10) 和 (2.11) 推得

$$\begin{aligned}\Delta u(k) =& d_1^T\left(G^TG+\lambda I\right)^{-1}G^T\left[Y_d(k)-Y_0(k)\right]\\ =& D_1^T\left[Y_d(k)-Y_0(k)\right],\end{aligned}$$

由式 (2.7) 得

$$Y_0(k) = H\Delta u(k) + Fy(k),$$

其中

$$H=\begin{bmatrix} z^{-1}H_1\\ z^{-2}H_2\\ \vdots\\ z^{-N}H_N\end{bmatrix}, F=\begin{bmatrix} F_1\\ F_2\\ \vdots\\ F_N\end{bmatrix},$$

由式 (2.9) 可得

$$Y_d(k) = C^*y(k) + \overline{C}^*y_r(k),$$

其中

$$C^*=\begin{bmatrix}\alpha\\ \alpha^2\\ \vdots\\ \alpha^N\end{bmatrix},\quad \overline{C}^*=\begin{bmatrix}1-\alpha\\ 1-\alpha^2\\ \vdots\\ 1-\alpha^N\end{bmatrix},$$

由前述三式可得

$$\Delta u(k)=D_1^T\left[\overline{C}^*y_r(k)-H\Delta u(k)+\left(C^*-F\right)y(k)\right],$$

考虑到实际系统输出 $y(k)$ 和模型输出 $y_m(k)$ 之差为 $e(k)$，

$$e(k)=y(k)-y_m(k),$$

$$y_m(k)=G_p\left(z^{-1}\right)u(k)=\frac{z^{-1}B\left(z^{-1}\right)}{A\left(z^{-1}\right)},$$

由前述三式可得

$$\begin{aligned}\Delta u(k)&=D_1^T\left[\overline{C}^*y_r(k)-H\Delta u(k)+\left(C^*-F\right)\left(G_p\left(z^{-1}\right)u(k)+e(k)\right)\right]\\&=D_1^T\overline{C}^*y_r(k)-D_1^TH\Delta u(k)+D_1^T\left(C^*-F\right)\left(G_p\left(z^{-1}\right)u(k)+e(k)\right)\end{aligned}$$

$$\begin{aligned}&\left(1+D_1^TH\right)\Delta u(k)+D_1^T\left(F-C^*\right)G_p\left(z^{-1}\right)u(k)\\&=D_1^T\overline{C}^*y_r(k)-D_1^T\left(F-C^*\right)e(k),\end{aligned}$$

$$\begin{aligned}u(k)&=\frac{D_1^T\overline{C}^*y_r(k)-D_1^T\left(F-C^*\right)e(k)}{\left(1+D_1^TH\right)\Delta+D_1^T\left(F-C^*\right)G_p\left(z^{-1}\right)}\\&=D_1^T\overline{C}^*y_r(k)-D_1^T\left(F-C^*\right)e(k)\frac{A\left(z^{-1}\right)}{\left(1+D_1^TH\right)\Delta A\left(z^{-1}\right)+D_1^T\left(F-C^*\right)z^{-1}B\left(z^{-1}\right)}\\&=\left[y_r-\frac{D_1^T\left(F-C^*\right)}{D_1^T\overline{C}^*}e(k)\right]G_C\left(z^{-1}\right)\\&=\left[y_r-G_F\left(z^{-1}\right)e(k)\right]G_C\left(z^{-1}\right)\end{aligned}$$

这里，各环节的离散传递函数分别为：

1. 预测模型：

$$G_P\left(z^{-1}\right)=\frac{z^{-1}B\left(z^{-1}\right)}{A\left(z^{-1}\right)},$$

2. 控制器：

$$G_C\left(z^{-1}\right)=\frac{D_1^T\overline{C}^*A\left(z^{-1}\right)}{\left(1+D_1^TH\right)\left(1-z^{-1}\right)A\left(z^{-1}\right)+D_1^T\left(F-C^*\right)z^{-1}B\left(z^{-1}\right)},$$

3. 滤波器：

$$G_F\left(z^{-1}\right)=\frac{D_1^T\left(F-C^*\right)}{D_1^T\overline{C}^*}.$$

由此可得到 GPC 系统的内模结构如图2.7所示。

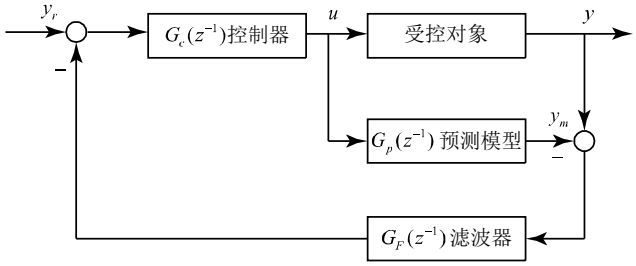

Figure 2.7: GPC 系统的内框结构图

2.4.7　GPC 的特性评价

主要优点

1. GPC 中采用 CARIMA 模型作为预测模型。该模型比较接近实际对象特性，而且具有积分作用，能有效清除静态误差，参数数目少，适合在线实现，对模型阶次不甚敏感。
2. 引入丢番图方程，并采用递推算法求解，节省了计算时间。
3. 采用有限时域的长时段多步预测，使 GPC 适用带负载扰动、未知或变滞后的被控对象。
4. 采用对输出误差和控制增量加权的二次型性能指标, 有利于提高闭环系统的稳定性。引入控制时域的概念，减小了计算量。

缺点

1. 仅适合于缓慢变化的系统，并且，跟踪轨迹也不能剧烈变化，若跟踪轨迹的变化速率过快，则跟踪效果较差。

2.4.8　GPC 的改进

1. 综合加权广义预测控制 (RSGPC)[1]

$$J=\sum_{j=1}^{N}\left[P\left(z^{-1}\right)\hat{y}(k+j|k)-R\left(z^{-1}\right)y_d(k+j)\right]^2+\lambda\sum_{j=1}^{N_u}\left[Q\left(z^{-1}\right)u(k+j-1)\right]^2$$

2. β-GPC[2]

$$\Delta u(k)=\beta d_1^T\left(G^TG+\lambda I\right)^{-1}G^T\left[Y_d(k)-Y_0(k)\right]$$

3. 比例积分型 GPC（PIGPC）[3]

$$\begin{aligned} J =& K_P \sum_{j=1}^{N} \left[\Delta\hat{y}(k+j|k) - \Delta y_d(k+j)\right]^2 + K_I \sum_{j=1}^{N} \left[\hat{y}(k+j|k) - y_d(k+j)\right]^2 \\ &+ K_D \sum_{j=1}^{N_n} [\Delta u(k+j-1)]^2 \end{aligned}$$

4. 极点配置 GPC（GPP）[4]

Bibliography

[1] 陈增强, 袁著祉, 李玉梅, et al. 工业锅炉的加权预测自校正控制 [J]. 自动化学报, 1993, 19(1):46-53.

[2] 孙明玮, 陈增强, 袁著祉. β 增量型广义预测控制 [J]. 控制理论与应用, 2000, 17(2):165-168.

[3] 陈增强, 车海平. 具有比例积分结构的广义预测自校正控制器 [J]. 控制与决策, 1994, 9(2): 105-110.

[4] M. A.LELIÄ, M. B.ZARROP. Generalized pole-placement self-tuning controller Part 1, Basic algorithm[J]. International Journal of Control, 1987, 46(2):547-568.

Appendix C Tracking Trajectory

跟踪轨迹的选择为

$$\begin{aligned} y_d(k) &= y(k), \\ y_d(k+i) &= \alpha y_d(k+j-1) + (1-\alpha) y_r(k), j = 1, 2, \cdots, N. \end{aligned}$$

为了求解最终轨迹的表达式，令

$$\begin{aligned} a[0] &= a_0, \\ a[k+1] &= \alpha a[k] + (1-\alpha) a_r. \end{aligned}$$

采用迭代方法可以得到

$$\begin{aligned} a[k+n] &= \alpha a[k+n-1] + (1-\alpha) a_r \\ &= \alpha \left[\alpha a[k+n-2] + (1-\alpha) a_r\right] + (1-\alpha) a_r \\ &= \alpha^2 a[k+n-2] + (1-\alpha) a_r \left(\alpha + 1\right) \\ &\cdots \\ &= \alpha^n a[k] + (1-\alpha) a_r \left(\alpha^{n-1} + \alpha^{n-2} + \cdots + \alpha + 1\right) \\ &= \alpha^n a[k] + \left(1-\alpha^n\right) a_r. \end{aligned}$$

即，跟踪轨迹在初始时刻 $a[k]$ 与目标值 a_r 之间使用指数柔化。