作业说明:

1. matlab 文件下的 hw1_1.m、hw1_2.m 分别对应课件上的 homework 1.1、1.2.
2. 本次 matlab 作业中采用的是 many relative timeline，此时分段的轨迹表达式为下式:

$$
f(t)=\begin{cases} f_1(t)\doteq\sum_{i=0}^{N} p_{1,i}t^i & 0\le t\le T_1-T_0=\Delta T_1 \\ f_2(t)\doteq\sum_{i=0}^{N} p_{2,i}t^i & 0\le t\le T_2-T_1=\Delta T_2 \\ \vdots & \vdots \\ f_M(t)\doteq\sum_{i=0}^{N} p_{M,i}t^i & 0\le t\le T_M-T_{M-1}=\Delta T_M \end{cases}
$$

3. 本次 matlab 作业的 Derivative constraints 规定为起始状态 pvaj、终止状态 pvaj，中间点 p。其中 p 值由输入点的坐标得到，起始、终止点处的 v、a、j 均设为 0;

```
start_cond = [waypoints(1), 0, 0, 0];
end_cond   = [waypoints(end), 0, 0, 0];
```

Continuity constraints 规定为前后两段轨迹交点处 pvaj 连续。

4. 本次作业独立求解 x 和 y 轴的多项式系数，该功能封装为函数 MinimumSnapQPSolver() 和 MinimumSnapCloseformSolver()，分别代表 QP 解法和闭式解法。

   4.1 对于MinimumSnapQPSolver()，需要实现函数体内调用的两个函数getQ()和 getAbeq()，实现可以参考课件。

$$f(t)=\sum_i p_i t^i$$

$$\Rightarrow f^{(4)}(t)=\sum_{i\ge4} i(i-1)(i-2)(i-3)t^{i-4}p_i$$

$$\Rightarrow \left(f^{(4)}(t)\right)^2=\sum_{i\ge4,l\ge4} i(i-1)(i-2)(i-3)l(l-1)(l-2)(l-3)t^{i+l-8}p_i\,p_l$$

$$\Rightarrow J(T)=\int_0^T \left(f^4(t)\right)^2 dt=\sum_{i\ge4,l\ge4}\frac{i(i-1)(i-2)(i-3)j(l-1)(l-2)(l-3)}{i+l-7}T^{i+l-7}p_i\,p_l$$

$$\Rightarrow J(T)=\int_0^T \left(f^4(t)\right)^2 dt=$$

$$\begin{bmatrix}\vdots\\ p_i\\ \vdots\end{bmatrix}^T\begin{bmatrix} & \vdots & \\ \cdots & \frac{i(i-1)(i-2)(i-3)l(l-1)(l-2)(l-3)}{i+l-7}T^{i+l-7} & \cdots \\ & \vdots & \end{bmatrix}\begin{bmatrix}\vdots\\ p_l\\ \vdots\end{bmatrix}$$

$$\Rightarrow \boxed{J_j(T)=\mathbf{p}_j^T\mathbf{Q}_j\mathbf{p}_j}$$ Minimize this!

- Derivative constraint for one polynomial segment
  - Also models waypoint constraint (0^{th} order derivative)

$$f_j^{(k)}(T_j)=x_j^{(k)}$$
$$\Rightarrow \sum_{i\ge k}\frac{i!}{(i-k)!}T_j^{i-k}p_{j,i}=x_{T,j}^{(k)}$$
$$\Rightarrow \begin{bmatrix}\cdots & \frac{i!}{(i-k)!}T_j^{i-k} & \cdots\end{bmatrix}\begin{bmatrix}\vdots\\ p_{j,i}\\ \vdots\end{bmatrix}=x_{T,j}^{(k)}$$
$$\Rightarrow \begin{bmatrix}\cdots & \frac{i!}{(i-k)!}T_{j-1}^{i-k} & \cdots\\ \cdots & \frac{i!}{(i-k)!}T_j^{i-k} & \cdots\end{bmatrix}\begin{bmatrix}\vdots\\ p_{j,i}\\ \vdots\end{bmatrix}=\begin{bmatrix}x_{0,j}^{(k)}\\ x_{T,j}^{(k)}\end{bmatrix}$$
$$\Rightarrow \mathbf{A}_j\mathbf{p}_j=\mathbf{d}_j$$

- Continuity constraint between two segments
  - Ensures continuity between trajectory segments when no specific derivatives are given

$$f_j^{(k)}(T_j)=f_{j+1}^{(k)}(T_j)$$
$$\Rightarrow \sum_{i\ge k}\frac{i!}{(i-k)!}T_j^{i-k}p_{j,i}-\sum_{l\ge k}\frac{l!}{(l-k)!}T_j^{l-k}p_{j+1,l}=0$$
$$\Rightarrow \begin{bmatrix}\cdots & \frac{i!}{(i-k)!}T_j^{i-k} & \cdots & -\frac{l!}{(l-k)!}T_j^{l-k} & \cdots\end{bmatrix}\begin{bmatrix}\vdots\\ p_{j,i}\\ \vdots\\ p_{j+1,l}\\ \vdots\end{bmatrix}=0$$
$$\Rightarrow \begin{bmatrix}\mathbf{A}_j & -\mathbf{A}_{j+1}\end{bmatrix}\begin{bmatrix}\mathbf{p}_j\\ \mathbf{p}_{j+1}\end{bmatrix}=0$$

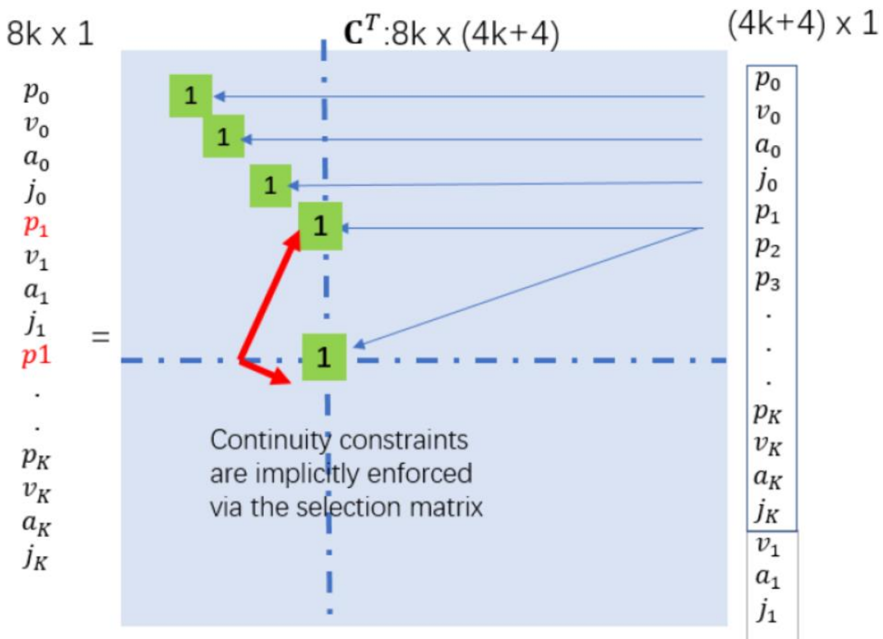
4.2 对于MinimumSnapCloseformSolver()， 需要实现函数体内调用的两个函数getM()和getCt()，根据自己的理解定义constrained variables dF。为了方便大家理解，下面给出一种构造方法。

假设采用7次多项式，一共有K段轨迹。$p_{i,k}$表示第k段多项式的第i个系数

$$\text{定义}P_{\mathrm{total}}=\begin{pmatrix}P_1\\ P_2\\ \vdots\\ P_M\end{pmatrix}^T=\begin{bmatrix}p_{0,1}\\ p_{1,1}\\ \cdots\\ p_{7,1}\\ \cdot\\ \cdot\\ \cdot\\ p_{0,K}\\ p_{1,K}\\ \cdots\\ p_{7,K}\end{bmatrix},\quad d_{\mathrm{total}}=\begin{bmatrix}d_1\\ d_2\\ \cdot\\ \cdot\\ \cdot\\ d_K\end{bmatrix}=\begin{bmatrix}p_0\\ v_0\\ a_0\\ j_0\\ p_1\\ v_1\\ a_1\\ j_1\\ \cdots\\ p_K\\ v_K\\ a_K\\ j_K\end{bmatrix},\quad \text{由}\mathrm{M_j}p_j=d_j\text{，可得}$$

$$\mathrm{p_{0,j}}=p_{j-1}$$
$$\mathrm{p_{1,j}}=v_{j-1}$$
$$2p_{2,j}=a_{j-1}$$
$$6p_{3,j}=j_{j-1}$$
$$p_{0,j}+p_{1,j}t_j+p_{2,j}t_j^2+p_{3,j}t_j^3+p_{4,j}t_j^4+p_{5,j}t_j^5+p_{6,j}t_j^6+p_{7,j}t_j^7=p_j$$
$$p_{1,j}+2p_{2,j}t_j+3p_{3,j}t_j^2+4p_{4,j}t_j^3+5p_{5,j}t_j^4+6p_{6,j}t_j^5+7p_{7,j}t_j^6=v_j$$
$$2p_{2,j}+6p_{3,j}t_j+12p_{4,j}t_j^2+20p_{5,j}t_j^3+30p_{6,j}t_j^4+42p_{7,j}t_j^5=a_j$$
$$6p_{3,j}+24p_{4,j}t_j+60p_{5,j}t_j^2+120p_{6,j}t_j^3+210p_{7,j}t_j^4=j_j$$

定义 $\mathrm{d_F} = \begin{bmatrix} p_0 \\ v_0 \\ a_0 \\ j_0 \\ p_1 \\ p_2 \\ \cdots \\ p_{K-1} \\ p_K \\ v_K \\ a_K \\ j_K \end{bmatrix}$， $\mathrm{d_P} = \begin{bmatrix} v_1 \\ a_1 \\ j_1 \\ \cdots \\ v_{K-1} \\ a_{K-1} \\ j_{K-1} \end{bmatrix}$， 由 $\begin{bmatrix} \mathbf{d}_1 \\ \vdots \\ \mathbf{d}_M \end{bmatrix} = \mathbf{C}^T \begin{bmatrix} \mathbf{d}_F \\ \mathbf{d}_P \end{bmatrix}$可得下列关系



5. 可能会用到的函数

5.1 blkdiag(a,b,c,d,⋯)

out = blkdiag(a,b,c,d,...)（其中 a、b、c、d、... 均为矩阵）输出以下形式的分块对角矩阵

$$\begin{bmatrix} a & 0 & 0 & 0 & 0 \\ 0 & b & 0 & 0 & 0 \\ 0 & 0 & c & 0 & 0 \\ 0 & 0 & 0 & d & 0 \\ 0 & 0 & 0 & 0 & \ldots \end{bmatrix}$$

输入矩阵不必是方阵，其大小也不必相等。

5.2 factorial(n)

返回n的阶乘

5.3 polyval(p,x)= p1xn + p2xn–1 + ··· + pnx + pn+1

返回在 x 处计算的 n 次多项式的值。输入参数 p 是长度为 n+1 的矢量，其元素是按要计算的多项式降幂排序的系数。

5.4 flipud(A)

返回一个相同长度的矢量，其元素的顺序颠倒。