Process Control-Design, Analysis and Assessment Doctoral Research Scholar Manikandan S Department of Chemical Engineering Indian Institute of Technology, Madras MATLAB Tutorial-Controller Design - Part 3

Welcome everyone to MATLAB tutorial for process control analysis, design and assessment. In this tutorial, we look at MATLAB implementation of model predictive controller. Just as an introduction, I will give the formulation of MPC and I will directly go to the code section of MATLAB and then show you how to code implement model predictive controller and how we can during the model predictive controller in order to get the performance we require and so on.

(Refer Slide Time 1:00)

MPC - Introduction

- MPC solves an optimization problem online
- Computes optimal input moves
- Can handle constraints
- · Easily extendable to MIMO systems
- Computationally time consuming



So the model predictive controller is a controller which solves an optimisation problem online. So the optimisation problem is formulated such that the model predictive controller uses model and et can be easily solved. Most of the time, we solve a quadratic program optimisation. It computes optimal input moves which is optimal in the sense of the objective function we have chosen. And since this is an optimisation problem, it can handle constraints easily.

And it is easily extendable to MIMO systems. The example I am going to show here is for SISO system. But the implementation is not restricted to see SISO system. It can be extended to MIMO. Only that we will have to look at how the structure of each term in the objective function

and prediction oriented terms are in MIMO systems. And MPC is also computationally timeconsuming but the examples which I am going to show here will be fast enough.

But if we have more inputs and more outputs for example like 25 cross 20 system where we have 25 out and we have 20 inputs then such systems will require more computational time in order to get the optimal in order to calculate the optimal solution for the optimisation.

(Refer Slide Time 2:55)

MPC - formulation $\min_{U} \sum_{i=1}^{P} (y^{sp}[k+i] - \hat{y}[k+i])^2 + \sum_{i=1}^{M} u[k+i-1]^2$
$$\begin{split} \hat{y}[k+i] &= h_1 u[k+i-1] + h_2 u[k+i-2] + \dots + h_\gamma u[k+i-\gamma] \quad \forall \, i \in (1,P) \\ u[k+i] &= u[k+M-1] \quad \forall \, i \in (M,P) \\ u^{LL} &\leq u[k+i] \leq u^{UL} \, \forall \, i \in (1,M) \end{split}$$

Now this is the formulation where we minimise the objective function where Y sat pine minus Y hat of K plus whole square and there is a minimum move objective added to the deviation from setpoint. Now the Y hat of K plus I is computed as an impulse response that finite table's response model with gamma coefficients. And we have the constraint that U of K plus I equals to U of K plus M minus 1 where I is M to P. So what we are essentially saying is though we have gamma elements of U to optimise, we choose to optimise only M elements.

And on top of it, we have limit constraints on U for all time instants which is from 1 to M. Notice that I have not specified what P is and what M is and neither have I specified the limits. So these will depend on individual applications and we may have to modify them to suit to get a better performance. Look at how we can compute and make the computation easier for us to implement it in MATLAB.

(Refer Slide Time 4:37)

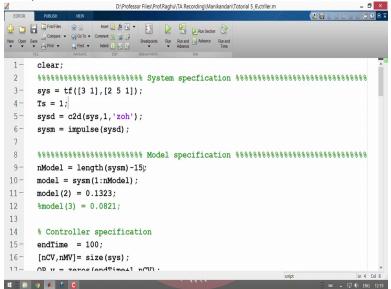
U.E (u(k+M)= u(k+M-) E

So we are computing summation I equals to 1 to P y of k plus i reference minus y hat of k plus i whole square. So we need to compute y of k plus I y hat of k plus i as a function of h1 to h gamma. So we have h1 u of k plus h2 u of k minus 1 all the way up to h gamma u of k minus gamma plus 1. Now in this optimisation function, we have minimized u. This u is constructed such that it has uk, uk plus 1 all the way up to uk plus m minus 1. So these are the unknowns which we which we have to optimise.

So this uk is unknown, whereas the rest of the terms are known. So this essentially evaluates to a constant. And the next term y of k plus 2 will be h1 times u of k plus 1 plus h2 times u of k plus all the way up to h gamma times u of k minus gamma plus 2. This will continue up to y hat of k plus m minus 1 which is h1 times u of k plus m minus 2 all the way up to h gamma times u of k minus gamma plus and minus 1. And y hat of k plus m is h1 times u of k plus m minus 1 all the way.

Similarly they will have y hat of k plus m plus 1. Here is where because the constraint that u of k plus m equals to u of k plus m minus 1. Because of that we will have h1 plus h2 times u of k plus m minus 1 plus all the way. So finally we will have y hat of k plus p which is h1 to h into u of k plus m minus 1 and etc. So we can notice the general structure where we have h1, h2 h1, h3 h2 h1, like this times u plus some known matrix which we can formulate.

(Refer Slide Time 8:17)



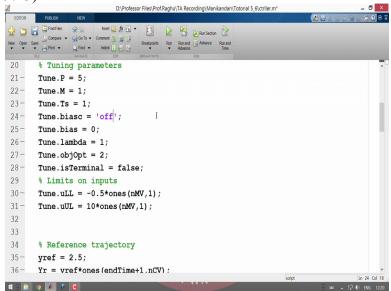
We will look at MPC implementation in MATLAB. Shown here is the code which implements an MPC for a SISO system. First I have declared the system here as a transfer function with the 3S plus 1 divided by 2S square plus 5S plus 1 as the transfer function. And I have discretised the system using a sampling time of 1 second. So TS equal to 1 implies that I have sampled the system once, for one second and I have computed the impulse coefficient using the inbuilt function, Impulse.

(Refer Slide Time 9:01)

2	D\\Professor Files\Prof.Raghu\TA Recording\Manikandan\Totorial 5_6\ctrller.m* = 🗗	×
EDITOR	PURUSH VEW	07
New Open S	Image: Derror Parks Image: Derror P	
5-	sysd = c2d(sys, 1, 'zoh');	^
6 -	<pre>sysm = impulse(sysd);</pre>	÷
7		
8	%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%	
9 -	nModel = length(sysm);	
10-	<pre>model = sysm(1:nModel);</pre>	
11-	<pre>%model(2) = 0.1323;</pre>	
12	<pre>%model(3) = 0.0821;</pre>	
13		
14	<pre>% Controller specification</pre>	
15-	endTime = 100;	
16-	[nCV,nMV]= size(sys);	
17-	OP.y = zeros (endTime+1, nCV);	
18-	OP.u = zeros(endTime+1,nMV);	
19		
20	% Tuning parameters	
21 -	Tune.P = 5:	v
e = 😭	scipt In 11 (-

Now in the next section, I have declared the model. So as a start we will start with the same system as the model and see how the performance is. Then we will look at introducing plant model mismatch and see how the MPC performs. So I will just comment the this line also.

(Refer Slide Time 9:23)

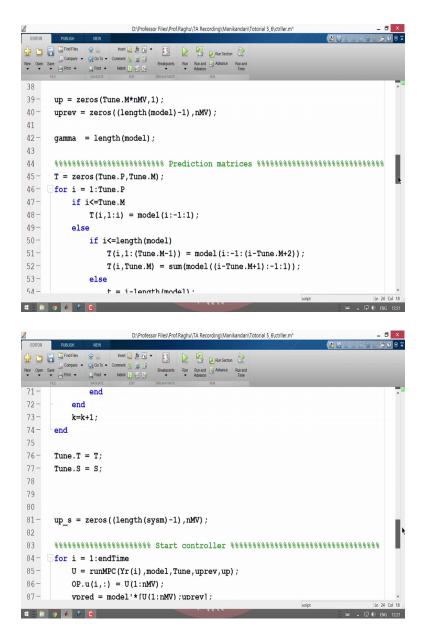


Then I move on to control specification where I specify the length of the simulation as 100 seconds and we initialise the output parameters y and u. Then I move on to tuning parameters which we can change based on the requirement. So the tuned at P is the prediction horizon, tuned at N is the control horizon. TS is again sapling time. Bias and biasc are bias correction parameters. we will start with dim off now and we will see how they affect the controller performance later.

Lambda is the multiplier used for waiting the objectives which is deviation from setpoint square plus minimum move. So the subjective option helps us decide which objective to be optimised in the MPC formulation.

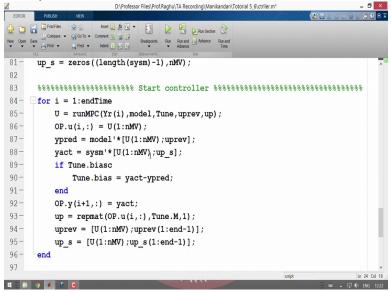
(Refer Slide Time 10:33)

	_ 8 ×
	()) (1 = 4 = 1 = 2 = 2 (0 = 1
🚽 🖿 🔚 🖓 Find Files 🕼 😒 Inset 🗟 🍂 🖉 💌 🔢 🔛 😼 🔛 Run Section 🕜	
New Open Sare Print V Find V Index 2 Print V Advance Run and V Run Advance Run Advance Run Advance Run Advance Run Advance Run Advance Run A	
FLE INVIGUTE EDT BREMPONTS RUN	-
27 - Tune.objOpt = 2;	^
<pre>28 - Tune.isTerminal = false;</pre>	
29 % Limits on inputs	
30 - Tune.uLL = -0.5*ones(nMV,1);	
31 - Tune.uUL = 10*ones(nMV,1);	
32	
33	-
34 % Reference trajectory	
35- yref = 2.5;	
<pre>36- Yr = yref*ones(endTime+1,nCV);</pre>	
<pre>37 %Yr = yref*(1-exp(-(1:endTime+1)/5));</pre>	
38	
39 - up = zeros (Tune.M*nMV, 1);	
40 - uprev = zeros((length(model)-1),nMV);	
41	
42 - gamma = length(model);	
4 D	Ln 24 Col 18
	□□ 🔺 🖓 🌵 ENG 12:20
D\\Professor Files\Prof.Raghu\TA Recording\Manikandan\Totorial 5_6\ctrller.m*	- 5 ×
EDTOR PUBLISH VEW	()) (1
Compare v Change (Compare V Con 2)	
New Open Sare C Pint + Q Find + Index 2 Sec. Sec. Run Runard Advance Runard Time	
THE INVITAGE EXIL DECAVOIRS NUT	
32	
32 33	
32 33 34 % Reference trajectory	×-
32 33 34 % Reference trajectory 35- yref = 2.5;	
<pre>32 33 34 % Reference trajectory 35- yref = 2.5; 36- Yr = yref*ones(endTime+1,nCV);</pre>	Î
<pre>32 33 34 % Reference trajectory 35- yref = 2.5; 36- Yr = yref*ones(endTime+1,nCV); 37 %Yr = yref*(1-exp(-(1:endTime+1)/5));</pre>	Ĩ
<pre>32 33 34 % Reference trajectory 35- yref = 2.5; 36- Yr = yref*ones(endTime+1,nCV); 37 %Yr = yref*(1-exp(-(1:endTime+1)/5)); 38</pre>	Ā
<pre>32 33 34 % Reference trajectory 35- yref = 2.5; 36- Yr = yref*ones(endTime+1,nCV); 37 %Yr = yref*(1-exp(-(1:endTime+1)/5)); 38 39- up = zeros(Tune.M*nMV,1);</pre>	ļ
<pre>32 33 34 % Reference trajectory 35- yref = 2.5; 36- Yr = yref*ones(endTime+1,nCV); 37 %Yr = yref*(1-exp(-(1:endTime+1)/5)); 38 39- up = zeros(Tune.M*nMV,1); 40- uprev = zeros((length(model)-1),nMV);</pre>	ļ
<pre>32 33 34 % Reference trajectory 35- yref = 2.5; 36- Yr = yref*ones(endTime+1,nCV); 37 %Yr = yref*(1-exp(-(1:endTime+1)/5)); 38 39- up = zeros(Tune.M*nMV,1); 40- uprev = zeros((length(model)-1),nMV); 41</pre>	ļ
<pre>32 33 34 % Reference trajectory 35- yref = 2.5; 36- Yr = yref*ones(endTime+1,nCV); 37 %Yr = yref*(1-exp(-(1:endTime+1)/5)); 38 39- up = zeros(Tune.M*nMV,1); 40- uprev = zeros((length(model)-1),nMV); 41 42- gamma = length(model);</pre>	l
<pre>32 33 34 % Reference trajectory 35- yref = 2.5; 36- Yr = yref*ones(endTime+1,nCV); 37 %Yr = yref*(1-exp(-(1:endTime+1)/5)); 38 39- up = zeros(Tune.M*nMV,1); 40- uprev = zeros((length(model)-1),nMV); 41 42- gamma = length(model); 43</pre>	ļ
<pre>32 33 34 % Reference trajectory 35- yref = 2.5; 36- Yr = yref*ones(endTime+1,nCV); 37 %Yr = yref*(1-exp(-(1:endTime+1)/5)); 38 39- up = zeros(Tune.M*nMV,1); 40- uprev = zeros((length(model)-1),nMV); 41 42- gamma = length(model); 43 44 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%</pre>	
<pre>32 33 34 % Reference trajectory 35- yref = 2.5; 36- Yr = yref*ones(endTime+1,nCV); 37 %Yr = yref*(1-exp(-(1:endTime+1)/5)); 38 39- up = zeros(Tune.M*nMV,1); 40- uprev = zeros((length(model)-1),nMV); 41 42- gamma = length(model); 43 44 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%</pre>	
<pre>32 33 34 % Reference trajectory 35- yref = 2.5; 36- Yr = yref*ones(endTime+1,nCV); 37 %Yr = yref*(1-exp(-(1:endTime+1)/5)); 38 39- up = zeros(Tune.M*nMV,1); 40- uprev = zeros((length(model)-1),nMV); 41 42- gamma = length(model); 43 44 %***********************************</pre>	
<pre>32 33 34 % Reference trajectory 35- yref = 2.5; 36- Yr = yref*ones(endTime+1,nCV); 37 %Yr = yref*(1-exp(-(1:endTime+1)/5)); 38 39- up = zeros(Tune.M*nMV,1); 40- uprev = zeros((length(model)-1),nMV); 41 42- gamma = length(model); 43 44 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%</pre>	



Now the then I have specified the lower and upper limits for input which are specified as minus 0.5 to 10. Then I declare reference trajectory which is the setpoint as of now as 2.5. And I declare such some internal variables. Now because we are using linear time invariant models, the matrices multiplying the decision variables uk to uk plus m minus 1 and the constant matrix are time invariant. Because of that, I precompute those matrices called T and S in this section and I store it in the structure tune.

(Refer Slide Time 11:24)



These are done apriory so that the computation time is reduced. Now runMPC is the functional which runs the optimisation for one iteration M provides solution from uk to uk plus m minus 1 and those values are stored in the variable u of which we will just select the first few first input variable, first-time instant input variables and apply it to the system. Notice that I have computed y predicted as well as y actual using model and system coefficients. This is to generalise the case where we plant model mismatch. So in case of plant model mismatch, predicted y and the actual y will be different.

So we will have to correct for that bias in a in real-time fashion. So that has to be updated in the MPC formulation.

(Refer Slide Time 12:31)

EDITOR PUBLISH							XX 🖪 🗟 🖌 🛛	i li 🦻 🖻 🛢 🕐
🐚 📑 🛄 Find Files	\$2	Inset 🛃 🏂 📲 🔹	13	Run Sec	ion			
Compare		Comment % 🎪 🏝	Breakpoints Run	Run and 🖳 Advance	Run and			
Plint •	NAVIGATE	EDIT	REAKPOINTS	Advance RUN	Time			
6 - end								
7								
8								
9- figure								
-		e,Yr,'r-')	;					
1- hold d								
	0:endTime							
	l('Time');						
4- ylabel	l('Outpu	t y');						
5- title	('Perform	mance of M	?C');					
6- legend	d('Refer	ence traje	ctory',	'System R	esponse');			
7		I						
8- figure	e(2);							
9- plot(0:endTime	e-1,0P.u(1	end-1)));				
0- xlabe	l('Time'));						
1- ylabe	l('Input	u');						
2- title	('Input)	<pre>profile');</pre>						
	APPS	EDITOR PUBLIS	H VEW	TLAB R2016a - acad	0	script	E C Search	Ln 24 Cc Ln 24 Cc Ln 24 Cc ENG Ln 24 Cc ENG Ln 24 Cc ENG Ln 24 Cc ENG Ln 24 Cc ENG Ln 24 Cc ENG Ln 24 Cc ENG
Compare	\$Q	Insert 🔜 fx 🚮 🔹 Comment % 🗽 🕄	H VEW	Run Sec	ion 🕑		E C Search	_ 12 () ENG _ 8
Find Files	\$Q	insert 🔜 🏂 🐔 🔹	H VEW	Run Sec	0		🚆 🖬	_ 12 () ENG _ 8
Open Save Print + FLE	 ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓<td>Inset 🛃 fx 🕺 🖌 Comment % 🗽 🖓 Indent 🛐 🕃 🚱 EDT nts + MATLA8</td><td>H VIEW</td><td>Run Ser Advance</td><td>ion D Run and Time</td><td></td><td>💥 🖬</td><td>_ 12 € ENG _ 13 Documentation</td>	Inset 🛃 fx 🕺 🖌 Comment % 🗽 🖓 Indent 🛐 🕃 🚱 EDT nts + MATLA8	H VIEW	Run Ser Advance	ion D Run and Time		💥 🖬	_ 12 € ENG _ 13 Documentation
Compare Open Save FILE TILE TILE TILE TILE TILE TILE TILE T		Inset 🛃 fx 🕺 🖌 Comment % 🗽 🖓 Indent 🛐 🕃 🚱 EDT nts + MATLA8	H VIEW	Run Ser Advance	ion D Run and Time		🚆 🖬	_ 12 () ENG _ 8
Open Save FLE Compare Pint + FLE Compare Pint + FLE Compare Pint + FLE Compare Pint + FLE Compare Pint + FLE Compare Pint + FLE Compare Pint + FLE		Inset 🛃 fx 🕺 🖌 Comment % 🗽 🖓 Indent 🛐 🕃 🚱 EDT nts + MATLA8	H VIEW	Run and Advance Advance Run andan\Totorial 5_6\	Run and Time		💥 🖼	_ 12 € ENG _ 13 Documentation
Open Save PLE PLE Compare PLE PLE Compare PLE PLE PLE PLE PLE PLE PLE PLE	 Go To + Go To + Find + Admin + Document Admin + Document Com × + funct: 	Inset S fr 1 - Comment % 2 3 Inder 7 5 6 Entr nts - MATLA8 iles/Prof.Raghu\TA Re ion U = run	H VEW	Ran and Advance R.N andan\Totorial 5.6\ ef, Model,	non De Run and Time unMPC.m		🚆 🖼	_ 12 € ENG _ 13 Documentation
Open Save Pint + FLE Pint + Pint +	 GoTo + G GoTo + G Find + MANGARE Admin + Document TOAProfessor Fi Famet: Gutta + Gutta + Gutta + Gutta + 	teset & fr ? • Comment % @ ? Inder ? • ? Inder	H VEW Breakports Run REAMOUNTS cording\Manika DMPC (yre De.uLL, 1	Ran See Ran and Advance RAN ef, Model, ' Fune. M, 1)	Rar and Time unMPCm Cune, OP, u0		in ar	_ 12 € ENG _ 13 Documentation
Open Save FEE C → Users TEE C → U	<pre>> GoTo + G Q Find + NAMARE > Admin + Docume r - D\Professor Fi Com × + Gunct: Ull = Ull = Uul =</pre>	toot : fr ? - Connert % * ?) noter : ? ? ? not : NAILA8 iles/Prof.Raghu/LA Re ion U = run repmat (Tun repmat (Tun	H VEW	Ram and Advance Ram and Advance Ram Ram Ram Ram Ram Ram Ram Ram Ram Ram	unMPC.m Time Tune, OP, u0)		_ 12 € ENG _ 13 Documentation
Compare Open Save	<pre>> GoTo + G GoTo + G Prod + PAdmin + Document - DAProfessor F PCm × + - funct: Ull = Ull = Ull = opt =</pre>	toot : fr ? - Connert % * ?) noter : ? ? ? not : NAILA8 iles/Prof.Raghu/LA Re ion U = run repmat (Tun repmat (Tun	H VEW	Ram and Advance Ram and Advance Ram Ram Ram Ram Ram Ram Ram Ram Ram Ram	unMPC.m Time Tune, OP, u0			_ 12 € ENG _ 13 Documentation
Compare Com	<pre>> define ></pre>	<pre>test & fr Comment % @ not ion U = run repmat(Tun repmat(Tun optimset(ne.isTermin</pre>	H VEW	Ram and Advance Ram and Advance Ram Ram Ram Ram Ram Ram Ram Ram Ram Ram	unMPC.m Time Tune, OP, u0)		_ 12 € ENG _ 13 Documentation
Image: State	<pre>* @ @ * @ @ * @ @ * @ @ * @ @ * @ # # # #</pre>	<pre>test Connect % Connec</pre>	H VEW	Advance Ran and Advance Ran and Advance Ran andan/Totorial 5.6V ef, Model, Fune.M, 1) Fune.M, 1) Fune.M, 1)	unMPC m Tune , OP , u0 ; I ; DIX', 1e-6,)		_ 12 € ENG _ 13 Documentation
Corper Open	 Admin + Document Admin + Document Admin + Document Admin + Gunet: Ull = Ull = Uul = opt = if Tuu Aeq = beq = 	<pre>test Connect % Connec</pre>	Breakports Ran RECARCONS RECARC	Rest Rest Rest Adaptic Adaptic Adaptic ext ext andam/Totorial 5.60 f f.Model,' f Fune.M,1) fune.M,1) rune.M,1) (end,:)*00	n De Read Read Tune, OP, u0 : I : : : : : : : : : : : : : :	(Lisplay',	Iter');	P 41 ENG Occumentation
C Find Files C Find Files Concess Co	 Admin + Document Admin + Document Admin + Document Conversion of the second second	<pre>test Connect % Connec</pre>	Breakports Ran RECARCONS RECARC	Rest Rest Rest Adaptic Adaptic Adaptic ext ext andam/Totorial 5.60 f f.Model,' f Fune.M,1) fune.M,1) rune.M,1) (end,:)*00	n De Read Read Tune, OP, u0 : I : : : : : : : : : : : : : :)	Iter');	P 41 ENG Occumentation
Corper Open Sue Phit	<pre>> Q Goto - Q Goto - > Admin + Documer = DAPOIdesson f = funct: Ull = Ull = Ull = Opt = if Tun Aeq = beq = [U,~,' else</pre>	<pre>test Connect % Connec</pre>	ti vew Breakonts Re- Resources AMPC (yree he.uLL, 1 he.uUL, 1 TolFun hal i, :) ; -Tune.S (hcon (@ (I	Ranse Ranse Ranse Advance Ranse Advance Ranse Ranse Rans Ranse	m Rand Tane Tune, OP, u0 : : : : : : : : : : : : : : : : : : :	<pre>(A Company) 'Display',' ne,yref,OP)</pre>	Iter'); ,u0,[],[- 0 x
Image: State	<pre>> gents - gents - and - and - and - and - gents - and - gents - gents</pre>	<pre>test Connect % Connec</pre>	ti vew Breakonts Re- Resources AMPC (yree he.uLL, 1 he.uUL, 1 TolFun hal i, :) ; -Tune.S (hcon (@ (I	Ranse Ranse Ranse Advance Ranse Advance Ranse Ranse Rans Ranse	m Rand Tane Tune, OP, u0 : : : : : : : : : : : : : : : : : : :	(Lisplay',	Iter'); ,u0,[],[- 0 x
Image: State of the	<pre>> Q Goto + Q Goto + Andre * Andre * DAProfessor F Gunet: Ull = Uul = opt = if Tuut Aeq = beq = [U,~,* else [u end</pre>	<pre>test Connect % Connec</pre>	ti vew Breakonts Re- Resources AMPC (yree he.uLL, 1 he.uUL, 1 TolFun hal i, :) ; -Tune.S (hcon (@ (I	Ranse Ranse Ranse Advance Ranse Advance Ranse Ranse Rans Ranse	m Rand Tane Tune, OP, u0 : : : : : : : : : : : : : : : : : : :	<pre>(A Company) 'Display',' ne,yref,OP)</pre>	Iter'); ,u0,[],[- 0 x
Image: Serie Compare Image: Compare Image: Compare Image: Compare	<pre>> definit + Cocome > Admin + Document > funct: Ull = Uul = Opt = if Tuu Aeq = beq = [U,~; else [U,~; end</pre>	<pre>test Connect % Connec</pre>	and the second s	Ranse Ranse Ranse Advance Ranse Advance Ranse Ranse Rans Ranse	m Rand Tane Tune, OP, u0 : : : : : : : : : : : : : : : : : : :	<pre>(A compare to a compare to</pre>	Iter'); ,u0,[],[- 0 x
Image: State of the	<pre>> definit + Cocome > Admin + Document > funct: Ull = Uul = Opt = if Tuu Aeq = beq = [U,~; else [U,~; end</pre>	<pre>test Q fr Q - Comment % @ @ @ hoter U = C = ion U = run repmat(Tuu repmat(Tuu repmat(Tuu repmat(Tuu repmat(Tuu repmat(Tuu repmat(1, :)) ~,~] = fmin U,~,~,~] =</pre>	and the second s	Ranse Ranse Ranse Advance Ranse Advance Ranse Ranse Rans Ranse	m Rand Tane Tune, OP, u0 : : : : : : : : : : : : : : : : : : :	<pre>(A compare to a compare to</pre>	Iter'); ,u0,[],[- 0 x
Image: State of the	<pre>> define +</pre>	<pre>test Q fr Q - Comment % @ @ @ hoter U = C = ion U = run repmat(Tuu repmat(Tuu repmat(Tuu repmat(Tuu repmat(Tuu repmat(Tuu repmat(1, :)) ~,~] = fmin U,~,~,~] =</pre>	and the second s	Ranse Ranse Ranse Advance Ranse Advance Ranse Ranse Rans Ranse	m Rand Tane Tune, OP, u0 : : : : : : : : : : : : : : : : : : :	<pre>(A compare to a compare to</pre>	Iter'); ,u0,[],[- 0 x

So in order to do that, I have computed both predicted y and the actual y. The final section of the code consists of plotting the results and correspondingly I have named the trajectories and given labels for y, x and titles. Let us look at how runMPC is configured so that we are sure about the coding. So here, you can see the runMPC function. It takes reference y, the model parameter as model, tuned structure as tuned, OP is the previous iteration previous time instant input variable.

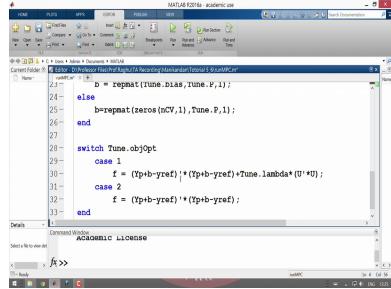
These are used to compute the constant which is part of MPC formulation and the u not is the initial condition or initial guess for the optimisation. In the first 2 lines, I have created a matrix

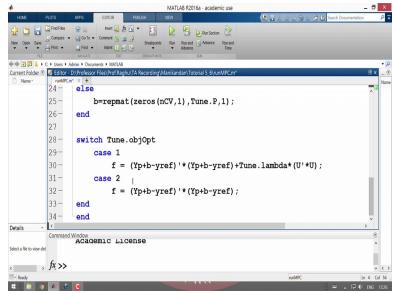
containing the input lower limit and upper limits repeated m times. In order to do that, I have used MATLAB function, repmat.

(Refer Slide Time 14:07) MATLAB R2016a - academic use Insert 🔜 🏂 🐔 🔹 Run Section 🚱 Go To 👻 Comment % 🤮 🎒 New Open Save Run ← → 🕃 💯 👗 → C: → Users → Admin → Documents → MATLAB Current Folder 💿 📝 Editor - D:\Professor Files\Prof.Rag Name Aeq - rune.r(ena,:); 7beq = yref(1,:)-Tune.S(end,:)*OP; [U,~,~,~] = fmincon(@(U)MPCobj(U,Model,Tune,yref,OP),u0,[],[],Aeq,} 8-9else [U,~,~,~] = fmincon(@(U)MPCobj(U,Model,Tune,yref,OP),u0,[],[], 10end 12 13 14end 15 16 function f = MPCobj(U,Model,Tune,yref,OP) < Command Window
ACademic License , *fx* >> 12 € ENG 12:24

And I have also given a provision for specifying terminal constraint which is u of k plus p equals to y setpoint which we will do in a later case. So now we will be running this case where we have no constraints other than limits.

(Refer Slide Time 14:35)

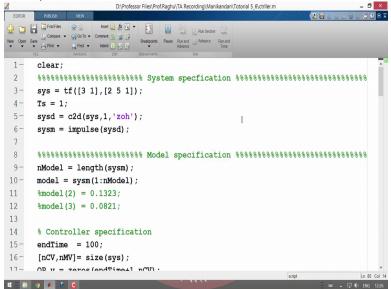


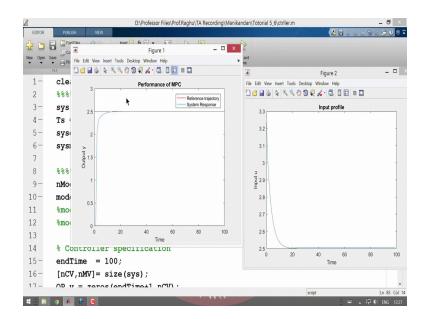


Now this MPC objective function computes the MPC objective which is y minus y setpoint whole square times whole square summation y minus y setpoint whole square for I equals to k plus 1 to k plus p. And similarly we have summation u of k plus i whole square where i runs from 0 to m minus 1. So that is calculated here. In order to compute yp, we use that matrices which we computed as T and S here. T times u. u is the decision variables containing u of k to u of k plus m minus 1 ad OP is the variable containing previous iteration u values.

So we will have YP as a vector of P output variables and then we update the bias here by constructing a vector of P values repeated either as 0s or using the constant which is computed in the previous iteration. Then we have 2 cases where in one case, we do not have this summation u u of k plus i whole square and in the other case where we have summation u of k plus i whole square, the effect of this will be easily seen later.

(Refer Slide Time 16:28)



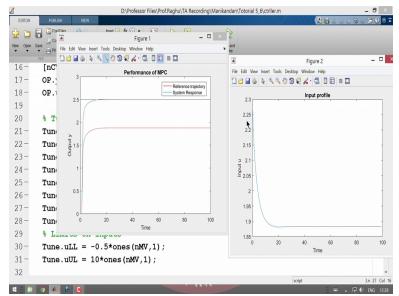


2		D:\Professor Files\Prof.Raghu\TA Recording\	Manikandan\Totorial 5_6\ctrller.m		
EDITOR	PUBU				i 🕤 🖄 🛱 🕐 🔍 🗖
🕂 🗅	E Co	Flee Charles Inseet C for Figure 1 - C	×		
New Open		File Edit View Insert Tools Desktop Window Help	and ne		
	FILE	1 🖆 🖬 🔌 💊 🔇 🏷 🐺 🔏 🛯 🗮 🗉 🖬	•	Figure 2	_ 0
1-	clea	Performance of MPC	File Edit View Insert Tool		
2	888	25	1 🖆 🖬 🎍 🔖 🔍 🍳	*****	1
3 -	sys	2.3	3.3	Input profile	
4 -	Ts :	2			
5 -	sys	2 Development of the second se	3.2		1
6 -	sysi		3.1		
7		入 1.5 - 1	3 -		
8	***				
9 -	nMo	1.	n 2.9 -		
10-	mode	0.5 -	2.8 -		
11	8mo	0.5	2.7 -		
12	8mo				
13		0 5 10 15 Time	2.6		
14	% Co	ntroller specification	2.5 0 20	0 40 60	80 100
15-	endI	ime = 100;	0 20	J 40 60 Time	00 100
16-	[nCV	/,nMV]= size(sys);			
17 -	00 1	= zeros (and Time 1 n (71) .			v
	0 4			script	Ln 85 Col 14

Let us move on and run the code. So this is the function that has to be studied. This is the script we have to run in order to see the performance of MPC. Let us run it. So now you can see that the system reaches the setpoint fairly quickly, within see 10 to 15 seconds. We can zoom in and see. So around 15 second it reaches to 0.5 and stays there. Notice the input variation. It starts from 3.3, goes down to 2.5. So it has 0.8 as the variation.

(Refer Slide Time 17:28)

Construction Particle West Construction Constructio	2	D:\Professor Files\Prof.Raghu\TA Recording\Manikandan\Totorial 5_6\ctrller.m		- 8 ×
Image: Compare + Control +	EDITO	R PUBLISH VEW	N A B L b i 9 C	☐ ? ○ ⁻
<pre></pre>		In Save		
<pre>16- [nCV,nMV]= size(sys); 17- OP.y = zeros(endTime+1,nCV); 18- OP.u = zeros(endTime+1,nMV); 19 20 % Tuning parameters 21- Tune.P = 5; 22- Tune.M = 1; 23- Tune.Ts = 1; 24- Tune.biasc = 'off'; 25- Tune.bias = 0; 26- Tune.lambda = 1; 27- Tune.objOpt = 2; 28- Tune.isTerminal = false; 29 % Limits on inputs 30- Tune.uLL = -0.5*ones(nMV,1); 31- Tune.uUL = 10*ones(nMV,1); 32</pre>	• •	🔹 🚔 Phint 👻 🖳 Find 💌 Indent 👔 🛃 🚱 🔹 💌 Advance 🦳 Time		
<pre>18- OP.u = zeros(endTime+1,nMV); 19 20 % Tuning parameters 21- Tune.P = 5; 22- Tune.M = 1; 23- Tune.Ts = 1; 24- Tune.biasc = 'off'; 25- Tune.bias = 0; 26- Tune.lambda = 1; 27- Tune.objOpt = 2; 28- Tune.isTerminal = false; 29 % Limits on inputs 30- Tune.uLL = -0.5*ones(nMV,1); 31- Tune.uUL = 10*ones(nMV,1); 32</pre>	16-			^
<pre>19 20 % Tuning parameters 21- Tune.P = 5; 22- Tune.M = 1; 23- Tune.Ts = 1; 24- Tune.biasc = 'off'; 25- Tune.bias = 0; 26- Tune.lambda = 1; 27- Tune.objOpt = 2; 28- Tune.isTerminal = false; 29 % Limits on inputs 30- Tune.uLL = -0.5*ones(nMV,1); 31- Tune.uUL = 10*ones(nMV,1); 32</pre>	17-	OP.y = zeros (endTime+1, nCV);		
<pre>20 % Tuning parameters 21- Tune.P = 5; 22- Tune.M = 1; 23- Tune.Ts = 1; 24- Tune.biasc = 'off'; 25- Tune.bias = 0; 26- Tune.lambda = 1; 27- Tune.objOpt = 2; 28- Tune.isTerminal = false; 29 % Limits on inputs 30- Tune.uLL = -0.5*ones(nMV,1); 31- Tune.uUL = 10*ones(nMV,1); 32</pre>	18-	OP.u = zeros(endTime+1,nMV);		
<pre>21- Tune.P = 5; 22- Tune.M = 1; 23- Tune.Ts = 1; 24- Tune.biasc = 'off'; 25- Tune.bias = 0; 26- Tune.lambda = 1; 27- Tune.objOpt = 2; 28- Tune.isTerminal = false; 29 % Limits on inputs 30- Tune.uLL = -0.5*ones(nMV,1); 31- Tune.uUL = 10*ones(nMV,1); 32</pre>	19			- 1
<pre>22- Tune.M = 1; 23- Tune.Ts = 1; 24- Tune.biasc = 'off'; 25- Tune.bias = 0; 26- Tune.lambda = 1; 27- Tune.objOpt = 2; 28- Tune.isTerminal = false; 29 % Limits on inputs 30- Tune.uLL = -0.5*ones(nMV,1); 31- Tune.uUL = 10*ones(nMV,1); 32</pre>	20	% Tuning parameters		
<pre>23- Tune.Ts = 1; 24- Tune.biasc = 'off'; 25- Tune.bias = 0; 26- Tune.lambda = 1; 27- Tune.objOpt = 2; 28- Tune.isTerminal = false; 29 % Limits on inputs 30- Tune.uLL = -0.5*ones(nMV,1); 31- Tune.uUL = 10*ones(nMV,1); 32</pre>	21-	Tune.P = 5;		- 1
<pre>24 - Tune.biasc = 'off'; 25 - Tune.bias = 0; 26 - Tune.lambda = 1; 27 - Tune.objOpt = 2; 28 - Tune.isTerminal = false; 29 % Limits on inputs 30 - Tune.uLL = -0.5*ones(nMV,1); 31 - Tune.uUL = 10*ones(nMV,1); 32 - xept in 55 cd 14</pre>	22-	Tune.M = 1;		
<pre>25- Tune.bias = 0; 26- Tune.lambda = 1; 27- Tune.objOpt = 2; 28- Tune.isTerminal = false; 29 % Limits on inputs 30- Tune.uLL = -0.5*ones(nMV,1); 31- Tune.uUL = 10*ones(nMV,1); 32</pre>	23-	Tune.Ts = 1;		
<pre>26- Tune.lambda = 1; 27- Tune.objOpt = 2; 28- Tune.isTerminal = false; 29 % Limits on inputs 30- Tune.uLL = -0.5*ones(nMV,1); 31- Tune.uUL = 10*ones(nMV,1); 32</pre>	24-	<pre>Tune.biasc = 'off';</pre>		
<pre>27 - Tune.objOpt = 2; 28 - Tune.isTerminal = false; 29 % Limits on inputs 30 - Tune.uLL = -0.5*ones(nMV,1); 31 - Tune.uUL = 10*ones(nMV,1); 32</pre>	25-	Tune.bias = 0;		
28- Tune.isTerminal = false; 29 % Limits on inputs 30- Tune.uLL = -0.5*ones (nMV,1); 31- Tune.uUL = 10*ones (nMV,1); 32 sofet	26-	Tune.lambda = 1;		
<pre>29 % Limits on inputs 30- Tune.uLL = -0.5*ones(nMV,1); 31- Tune.uUL = 10*ones(nMV,1); 32 solpt [in 65 Col 14]</pre>	27 -			
30- Tune.uLL = -0.5*ones(nMV,1); 31- Tune.uUL = 10*ones(nMV,1); 32	28-	Tune.isTerminal = false;		
31- Tune.uUL = 10*ones(nMV,1); 32 solpt (in 65 Col 14	29	% Limits on inputs		
32 solpt (in 65 Cel 14	30-	Tune.uLL = -0.5 *ones(nMV,1);		
script In 85 Col 14	31-	Tune.uUL = $10 \times cnes(nMV, 1)$;		
	32			~
	-			



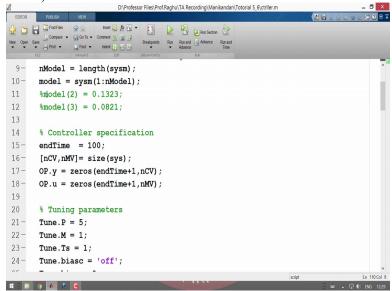
Now without closing this we window, I am going to just change the type of objective which we have used. So this, the objective we have used did not have the summation of u square. So if if we just change the objective, how will the system perform is what we are going to look at now. Now you can see that initially the whole variation was around 3.3 to 2.8 which was 0.5 and here also it is similar but the output since the initial input was 3.3 at that time, the output was able to reach 2.5. But since we have minimal variation in the input as a objective, there is a bias in the performance.

(Refer Slide Time 18:36)

2	D:\Professor Files\Prof.Raghu\TA Recording\Manikandan\Totorial 5_6\ctrller.m*		_ 8 ×
EDITOR	PUBLISH VEW		5 9 2 2 0
4 🖬	🔚 🕞 Find Files 🖉 🔄 Insert 🛃 🍂 🖓 🔹 📰 🔛 🔛 🔛 Run Sectors		
New Open 3	and Compare V Go to V Comment 70 20 20 Realmoints Run Renand C Advance Run and		
• •	Construction of the second secon		
96-	end		^ _
97			
98			
99-	figure(1);		
100-	<pre>plot(0:endTime,Yr,'r-');</pre>		
101-	hold on;		
102-	<pre>plot(0:endTime,OP.y);</pre>		
103-	<pre>xlabel('Time');</pre>		
104-	<pre>ylabel('Output y');</pre>		
105-	<pre>title('Performance of MPC');</pre>		
106-	<pre>legend('Reference trajectory','System Response');</pre>		
107			
108-	figure(2);		
109-	<pre>plot(0:endTime-1,OP.u(1:end-1));</pre>		
110			
111 -	<pre>xlabel('Time');</pre>		
112-	<pre>ylabel('Input u');</pre>		~
		script	Ln 110 Col 1
4 🗄 🎬	9 4 F E		🔺 🛱 🌓 ENG 12:28

So we can tune that by tuning this lambda. We will put minimal weight to the 2nd term which is summation u square. Let us see how that works. Now you can see the response actually reach the setpoint and the variation is like 3.3 to 2.5 which was the previous variation also.

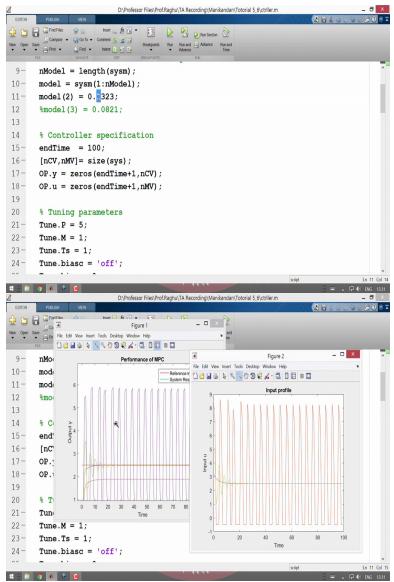
(Refer Slide Time 19:21)



Now let us look at introducing plant model mismatch and how the system performs. I will just change the 2nd parameter to say 0.1323. So what we are saying is all the parameters other than 2nd term in the model is estimated properly. This particular coefficient is estimated in a wrong way. So if we do that without the bias correction, how will the performance be? Now you can see, we have oscillatory performance. This oscillation is because of the constraint which is being forced in the object optimisation function. You can see that it becomes minus 0.5 for 2 seconds and then it goes back. So this is the reason why the response is oscillatory.

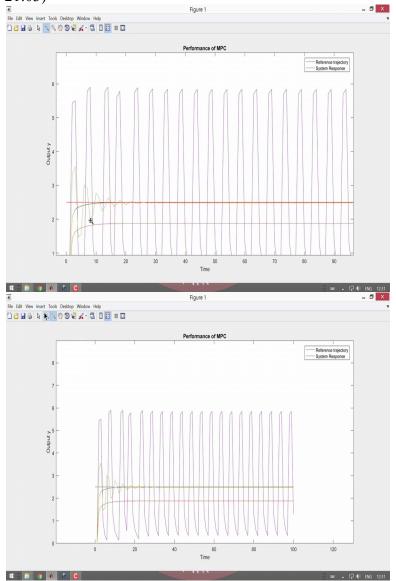
(Refer Slide Time 20:33)

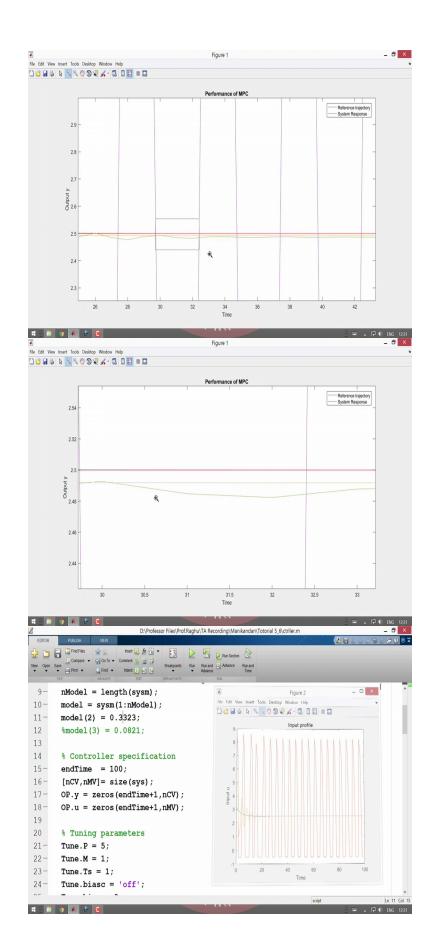
					P	ALLADIN	2016a - acad	emic use									- 8	x
	PLOTS	APPS	EDITOR	PUBLISH	H VIE	W		1.2.25		(48	4 b b	26	C 2	earch Do	ocumentation	1	P
	Find Files	\$ Q	Insert 🛃 f.			2 🛯	Run Sed	tion 😥										
	Compare 🔻	Go To 👻 Ci			Breakpoints	Run Runa	and 🖳 Advance	Run and										
FIE	Print 🔻		Indent 🛐 👳	5 1 2	PEAKDOWTS	 Adva 	Ince DLN	Time										
1 🗼 + C	D: • Professor Fi	les + Prof.Raghu	 TA Recordi 	ng ⊁ Manika	andan 🕨 Totori	al 5_6												Ŧ
older ⊙ e∗		:\Professor File	es\Prof.Rag	hu\TA Rec	ording\Mar	nikandan\	Totorial 5_6\	runMPC.m	n					0		Vorkspace		
)6.pptx	runMPC.m 24 -	else												-		lame ~	Value 100	5
5.pptx	25-		ronma	+ / 707	or InC	1	Tuno	D 11.							^	gamma	24	1)
m ISV			repilla	c(zer	os (nc	v,1),	, Tune . I	e,1);									100 23	1) 1)
ing_1.slx	26-	end													Ē	k	6	1)
decou	27															Nev Nev	24x1 do. 1	24
	28-	switch	Tune	.objC	pt											nModel nMV	24	1) 1)
	29-	ca	se 1													OP	1x1 struc	t 1x
	30-		f =	(Yp+	b-vre	f) '*	(Yp+b-y	vref)	+Tm	ne.1	Lamh	da* (1	זי *וז):		S sys	5x23 do. 1x1 tf	. 5x 1x
	31-		se 2		- 1-0	-,	·	/						. '		sysd sysm	1x1 tf 24x1 do.	1x
		Ca						-								T	(0;0.132.	. 5x
	32 -		f =	(Yp+	-b-yre	t)'*	(Yp+b-y	yref)	;							Ts Tune	1 1x1 struc	1) t 1)
	33-	end														U	-0.5000	1x
	34-	end													~	up_s	-0.5000 23x1 do.	
~	<													>		uprev	23x1 do. 1.2896	23 1x
	Command V	Vindow CCTLLET													•	ypred	1.5396	1x
ew det		trller														Yr yref	101x1 d. 2.5000	. 10 1x
		criter																
	$f_{x} >>$														~ <	(
🖉 Open ·	PLOTS Rows	Columns		VIEW	_	/ATLAB R	12016a - acad	lemic use		(4 🖬 ,	6 10 12	26	## # \$. I⊋ €	- 8	х
🖌 Open -	PLOTS Rows	Columns	I .		_	MATLAB R	12016a - acad	lemic use		(8 8 1	6 10 12	96	*** ***			- 8	х
Print		Columns in 1 in crion les > Prof.Raghu	sert Delete	Transpose						(1	6 10 2	223		iearch Do	ocumentation	- 8	2:30 ×
Copen - Print BLE Ider ()	PLOTS Rows 2 SELE C: Professor Fi C: Editor - n	Columns In 1 In cnow les Prof.Raghu unMPC.m	sert Delete	Transpose			2016a - acad			(A 11	6 6 1	26		iearch Do	vorkspace	- 8	× • •
Open → Print → C C C C C C C C C C C C C C C	Rows Rows Z SELE Yrofessor Fi Editor - rr model % 24x1 doubkr	Columns 1 In In COOM INCIDICATION IN COOM IN COOM INCIDICATION IN COOM INCIDICATION INCI	sert Delete	Transpose						(8		22		iearch Do	ocumentation	- 8	× • •
Cpen · Print · LE der ⑦ · Mo.pptx)5.pptx	Rows Z SELE Xet A A Rows Z SELE Xet A	Columns 1 In In COOM INCIDICATION IN COOM IN COOM INCIDICATION IN COOM INCIDICATION INCI	sert Delete	Transpose					13	14	15			(earch Do	ocumentation Vorkspace Jame ~	- 0	× P Si 1x
Open · Print · er · • .pptx .pptx	Rows Rows	Columns 1 In In COOM INCIDICATION IN COOM IN COOM INCIDICATION IN COOM INCIDICATION INCI	sert Delete EDIT • TA Recordi	G Y Transpose Sort ▼ ng ≯ Manika	andan + Totori	al 5_6	🖌 Variabl	es - sysm						(iearch Do	Vorkspace lame ~ endTime gamma i	- 0 Value 100 24 100 23	× • 5 10 10 10 10 10 10 10 10 10 10
Copen → Print E Ler ⑦	Rows Rows Rows Z SELE Selector Construction	Columns 1 In In COOM INCIDICATION IN COOM IN COOM INCIDICATION IN COOM INCIDICATION INCI	sent Delete EDIT • TA Recordi	G Y Transpose Sort ▼ ng ≯ Manika	andan + Totori	al 5_6	🖌 Variabl	es - sysm						(earch Do	Vorkspace lame - endTime j k	- 0	×
Open Print E der (*) 6.pptx 5.pptx n sv ng_1.stx	Rows 2 SELE Professor Fi Editor - rr model × 2 Mathematical Action of the second of th	Columns 1 In In COOM INCIDICATION IN COOM IN COOM INCIDICATION IN COOM INCIDICATION INCI	sent Delete EDIT • TA Recordi	G Y Transpose Sort ▼ ng ≯ Manika	andan + Totori	al 5_6	🖌 Variabl	es - sysm						(earch Do	Vorkspace lame - endTime gamma i k model nCV	- 0 Value 100 24 100 23 6 24x1 do. 1	× • • • • •
Open → Print er ⊙ 	Rows Rows Rows Z SELE Selector Construction	Columns 1 In In COOM INCIDICATION IN COOM IN COOM INCIDICATION IN COOM INCIDICATION INCI	sent Delete EDIT • TA Recordi	G Y Transpose Sort ▼ ng ≯ Manika	andan + Totori	al 5_6	🖌 Variabl	es - sysm						(earch Do	Vorkspace lame - endTime j k model	Value 100 24 100 23 6 24x1 do.	× S 10 10 10 10 10 10 10 10 10 10
Copen - Print E der (*) 6.pptx 5.pptx 5.pptx 9 sv ng_1.sk	A B PLOTS PLOTS V Rows 2 SELE SELE Professor Fi Zet Honded 2 October 2 3 0.1169 4 0.0539 5 0.0392 6 0.0311	Columns 1 In In COOM INCIDICATION IN COOM IN COOM INCIDICATION IN COOM INCIDICATION INCI	sent Delete EDIT • TA Recordi	G Y Transpose Sort ▼ ng ≯ Manika	andan + Totori	al 5_6	🖌 Variabl	es - sysm						(9	Vorkspace lame ~ endTime gamma i k model nCV MModel nMV OP	Value 100 24 100 23 6 24x1 do. 1 24 1 Tx1 struc	× × × × × × × 1:
Copen - Print E der (*) 6.pptx 5.pptx 5.pptx 9 sv ng_1.sk	Image: Arrows and Ar	Columns 1 In In COOM INCIDICATION IN COOM IN COOM INCIDICATION IN COOM INCIDICATION INCI	sent Delete EDIT • TA Recordi	G Y Transpose Sort ▼ ng ≯ Manika	andan + Totori	al 5_6	🖌 Variabl	es - sysm						(9 9	Vorkspace lame - endTime gamma i k model nCV nModel nMV OP S S sys	- 0 Value 24 100 23 6 6 24/1 da 1 1xl struc 5x32 da.	× S 1: 1: 1: 1: 1: 1: 1: 1: 1: 1:
Copen → Print E Ler ⑦	A R ELOTS SELE Comparison SELE SELE Professor Fi Comparison Comparison Comparison Compariso	Columns 1 In COLUMN COL	sent Delete EDIT • TA Recordi	G Y Transpose Sort ▼ ng ≯ Manika	andan + Totori	al 5_6	🖌 Variabl	es - sysm						(earch Do	Vorkspace lame - endTime j i k model nCV MModel nMV OP S S S S S S S S S S S S S S	- 0 Value 100 23 6 24/1 do. 1 1 24 100 23 6 24/1 do. 1 1 x1 struc 5x23 do. 1x1 tf 1x1 struc	× × ×
Open Print E der (*) 6.pptx 5.pptx n sv ng_1.stx	A P ROMS SELE * Podescor Fi Sele 2 Sele 3 0.1169 4 0.0559 5 0.03311 7 0.0248 8 0.0200 9 0.0161 10 0.0124	Columns 1 In COLUMN COL	sent Delete EDIT • TA Recordi	G Y Transpose Sort ▼ ng ≯ Manika	andan + Totori	al 5_6	🖌 Variabl	es - sysm						(earch Do	Vorkspace Jame - endTime girma i k model nAV p S S S S S S S S S S S S S S S S S S	- 0 Value 24 100 23 6 6 24/1 da 1 1xl struc 5x32 da.	×
Copen - Print E der (*) 6.pptx 5.pptx 5.pptx 9 sv ng_1.sk	A R Plots Rows 2 SELE 2 Professor Fi 2 Editor - n 2 Costa 3 0.1169 4 0.0539 5 0.0392 6 0.0311 9 0.0161 10 0.0129	Columns 1 In COLUMN COL	sent Delete EDIT • TA Recordi	G Y Transpose Sort ▼ ng ≯ Manika	andan + Totori	al 5_6	🖌 Variabl	es - sysm						(earch Do	Vorkspace endTime gamma j k model nAV sys sys sys sys	- 0 Value 100 24 100 24 1 124 1 124 1 124 1 124 1 124 1 124 1 124 1 124 1 124 1 124 1 124 1 124 1 124 1 124 1 124 1 124 1 124 12 12 12 12 12 12 12 12 12 12 12 12 12	×
Open Print E der (*) 6.pptx 5.pptx n sv ng_1.stx	A P Rows 2 SEE SEE 2 SEE 2 SEE 2 Modesor Fi 2 Modesor Fi 2 Modesor Fi 2 Modesor Fi 3 0.1169 1 0 2 0.0332 6 0.0311 10 0.0129 11 0.0104 12 0.0067 14 0.0354	Columns 1 In COLUMN COL	sent Delete EDIT • TA Recordi	G Y Transpose Sort ▼ ng ≯ Manika	andan + Totori	al 5_6	🖌 Variabl	es - sysm						(earch Do	Vorkspace Anne - endline samma i j k model nCV model nCV model nCV model sys Sys Sys Sys T T s fune U	- 0 Value 100 24 100 23 6 2441 do. 124 101 1x1 struct 102 11 1x1 struct - 0,5000	× × × × × × × × × × × × × × × × × × ×
Copen • Print RE der • D6.pptx D5.pptx m asv ing_1.sk ; decou	A Rous Cons	Columns 1 In COLUMN COL	sent Delete EDIT • TA Recordi	G Y Transpose Sort ▼ ng ≯ Manika	andan + Totori	al 5_6	🖌 Variabl	es - sysm						(P × V	Vorkspace anne - endTime - endTime - i j k model nMV oP S S syst opd syst T T Tune U up up,S	- C Value 100 24 100 23 6 24 100 23 6 24 11 101 102 24 100 124 100 100 23 6 24 11 101 102 103 104 105 105 105 105 105 105 105 105	× × ×
Print ABLE	Correction C	Columns di tanta di tan	sent Delete EDIT • TA Recordi	G Y Transpose Sort ▼ ng ≯ Manika	andan + Totori	al 5_6	🖌 Variabl	es - sysm						(vorkspace lame - endline gamma k model nCV oP S S S S S S S S S S S S S S S S S S	- C Value 100 24 100 24 100 23 6 24x1 do. 1 1 24 1 1 1 1 1 1 1 1 24x1 do. 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 4 1 1 1 2 4 1 1 1 2 4 1 1 1 2 4 1 1 2 3 3 6 1 1 2 4 1 1 2 3 3 6 1 1 2 4 1 1 2 3 3 6 1 1 2 4 1 1 2 3 3 6 1 1 2 4 1 1 2 3 3 6 1 1 2 4 1 1 2 3 3 6 1 1 2 4 1 1 2 3 3 6 1 1 2 4 1 1 2 3 3 6 1 1 2 4 1 1 2 3 3 6 1 1 1 1 2 3 3 6 1 1 1 2 3 3 6 1 1 1 1 2 3 3 6 1 1 1 2 3 3 6 1 1 1 1 2 3 3 6 1 1 1 1 2 3 3 6 1 1 1 1 2 3 3 6 1 1 1 1 2 3 3 6 1 1 2 3 3 6 1 1 1 2 3 3 6 1 1 2 3 3 6 1 1 1 1 1 2 3 3 6 1 1 2 3 3 6 1 1 2 3 5 3 5 3 5 3 5 3 5 1 1 1 2 3 5 3 5 3 5 1 1 1 2 3 5 1 1 1 2 3 5 3 5 3 5 3 5 1 1 2 3 5 1 1 1 2 3 5 1 1 1 2 3 5 1 1 1 1 2 3 5 3 5 1 1 1 2 3 5 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	× (((())) ())) ()))))))))))))
Casy Construction Constructi	Correction C	Columns 1 In COLUMN COL	sert Delete EDIT • TA Recordi	G Y Transpose Sort ▼ ng ≯ Manika	andan + Totori	al 5_6	🖌 Variabl	es - sysm						(Vorkspace Jame - gamma i j k model mModel mModel mModel syst S S S S S S S S S S S S S S S S S S S	- C Value 100 23 6 24 102 24 102 24 102 24 102 124 100 124 100 124 100 124 11 12 11 12 11 12 11 241 30 231 100 123 100 100 123 100 100 100 100 100 100 100 100 100 10	× (Sii 1x 1x 1x 1x 1x 1x 1x 1x 1x 1x
Open Print Print ILE Ider O6.pptx 05.pptx m asv ing_1.sk c_decou	Conservation	Columns di 1 In In crow les - Profaghu JMPC.m yosm II 2 3 1 2 3 1 4 1 1 1 1 1 1 1 1 1 1 1 1 1 1	sert Delete EDIT • TA Recordi	G Y Transpose Sort ▼ ng ≯ Manika	andan + Totori	al 5_6	🖌 Variabl	es - sysm						(Vorkspace Jame - endTime - endTime - endTime - gyrma i k model nCV mModel rCV mModel rCV S Syst Syst T S Tone U upp uppev uppev uppet	- 6 Value 100 24 100 23 6 24/1 do. 11 12/1 struc (0.132 1 12/1 fat (0.132) 1 12/1 fat (0.132) 1 12/2 fat (0.132) 1 12/2 1 12/2 1 12/2 12/2 1 12/2 12/2	× • • • • • • • • •
Copen + Print = Print = Copen + Copen + Cop	Conservation	Columns di tanta di tan	sert Delete EDIT • TA Recordi	G Y Transpose Sort ▼ ng ≯ Manika	andan + Totori	al 5_6	🖌 Variabl	es - sysm						(Vorkspace and the second secon	- 6 Value 100 24 100 23 6 24/1 do. 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	× • • • • • • • • •



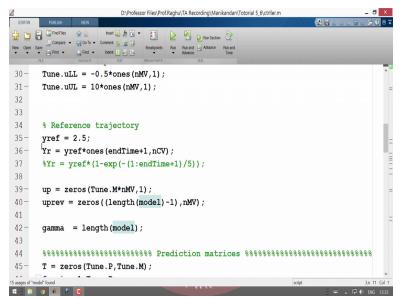
The actual model function of the 2^{nd} constant has the value of 0.6 sorry, 0.6323. So when the model is changed, the 2^{nd} coefficient is changed by 0.5, we have oscillatory response.

(Refer Slide Time 21:03)





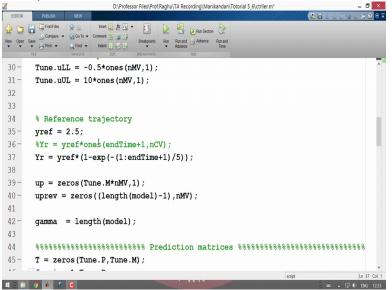
2	D:\Profes	sor Files\Prof.Raghu\TA Recording\M	Manikandan\Totorial 5_6\ctrller.m	_ 8 ×
EDITOR	PUBLISH VIEW			
🕂 🗅	Find Files Compare ▼ Go To ▼ Comment % So To ▼	' 🔢 📐 😤 🖉 Run Se	ction 🕑	
New Open	Save Pint + Q Find + Indent 1 - Te	Breakpoints Run Run and Advance	e Run and Time	
	FILE NAVIGATE EDIT	BREAKPOINTS RUN		
23-	Tune.Ts = 1;			^
24-	Tune.biasc = 'o <mark>ff</mark> ';			
25-	Tune.bias = 0;			
26-	Tune.lambda = $0.01;$			
27-	Tune.objOpt = 1;			
28-	Tune.isTerminal = fals	e;		
29	% Limits on inputs			
30-	Tune.uLL = -0.5 *ones(n)			
31-	Tune.uUL = 10*ones(nMV	,1);		
32				
33				
34	% Reference trajectory			
35-	<pre>yref = 2.5;</pre>			
36-	Yr = yref*ones(endTime			
37	%Yr = yref*(1-exp(-(1)))	endTime+1)/5));		
38				
30-	un = zoroe/Tuno M*nM7	11.		script In 11 Col 15
11 II 😤				
And in case of the local division of the loc	Di Brofor	cor Filer) Drof Paghu) TA Pacording) A	Aanikandan\Totorial 5 6\ctrllorm	
EDITOR	PUBLISH VIEW	sor Files\Prof.Raghu\TA Recording\A Figure 1	Manikandan\Totorial 5_6\ctrller.m	
EDITOR	PUBLISH VEW	sor Files\Prof.Raghu\TA Recording\N Figure 1 ew Insert Tools Desktop Window Help		_ 8 ×
2	PUBLISH VEW Image: Compare the second seco	Figure 1	× □ _	_ 8 ×
EDITOR	FUBLISH VEW □ Find Files Q □ Find Files Q □ Compare Q □ Compare Q □ File Edt U Compare Q □ File Edt	Figure 1 ew Insert Tools Desktop Window Help	×	_ 8 ×
EDITOR	FUBLISH VEW □ Find Files ♀ □ Find Files ♀ □ Compare ♀ □ Compare ♀ □ Pint ▲ Indext ↓ Indext ↓	Figure 1 ew Insert Tools Desktop Window Help	MPC Reference trajectory	_ 8 ×
EDITOR Control New Open	PuBLISH VEW □□ FindFiles □□ FindFiles □□ Compare □□ Compare □□ Find □□ Find □□ Find □□ Find □ Find □ Find □ Find □ Find □ Find □ Compare □ Find □ Find □ Find	Figure 1 ew Insert Tools Desktop Window Help	- C ×	_ 8 ×
EDITOR EDITOR New Open 23-	PLELOH VEW □ □	Figure 1 ew Insert Tools Desktop Window Help	MPC Reference trajectory	_ 8 ×
EDITOR EDITOR New Open 23- 24-	PLEACH VEW Image: Fill of the state of	Figure 1 ew Insert Tools Desktop Window Help	MPC Reference trajectory	_ 8 ×
EDITOR EDITOR New Open 23- 24- 25-	PLEACH VEW Image: Fill of the state of	Figure 1 ew Insert Tools Desktop Window Help	MPC Reference trajectory	_ 8 ×
23 - 24 - 25 - 26 -	PLECOF VEW Image: The second	Figure 1 ew Insert Tools Desktop Window Help	MPC Reference trajectory	_ 8 ×
EDITOR EDITOR New Open 23 - 24 - 25 - 26 - 27 -	PLECOH VEW Image: The second	Figure 1 ew Insert Tools Desktop Window Help	MPC Reference trajectory	_ 8 ×
23 - 24 - 25 - 26 - 27 - 28 -	PLECOV VEW Image: Comment of the state	Figure 1 ew Insert Tools Desktop Window Help	MPC Reference trajectory	_ 8 ×
23 - 24 - 25 - 26 - 27 - 28 - 29	PLEND# VEW Image: Figure 1 Image: Figure 1 Image: Figure 1 Figure 1 Image: Figure 1 Image: Figure 1 Image: Figure 1 Tune. Ts = 1; Tune. biasc = 0; 2557 Tune. biasc = 0; Tune. lambda = 0.(2505 Tune. lambda = 0.(Tune. objOpt = 1; 2405 Tune. uLL = -0.5* Tune. uLL = 245.* 2405	Figure 1 ew Insert Tools Desktop Window Help	MPC Reference trajectory	_ 8 ×
■ </td <td>Pless VEW Image: Filler Fil</td> <td>Figure 1 ew Insert Tools Desktop Window Help</td> <td>MPC Reference trajectory</td> <td>_ 8 ×</td>	Pless VEW Image: Filler Fil	Figure 1 ew Insert Tools Desktop Window Help	MPC Reference trajectory	_ 8 ×
EDITOR 23- 24- 25- 26- 27- 28- 29 30- 31-	PLEND# VEW Image: Figure 1 Image: Figure 1 Image: Figure 1 Figure 1 Image: Figure 1 Image: Figure 1 Image: Figure 1 Tune. Ts = 1; Tune. biasc = 0; 2557 Tune. biasc = 0; Tune. lambda = 0.(2505 Tune. lambda = 0.(Tune. objOpt = 1; 2405 Tune. uLL = -0.5* Tune. uLL = 245.* 2405	Figure 1 ew Insert Tools Desktop Window Help	MPC Reference trajectory	_ 8 ×
23 - 24 - 25 - 26 - 27 - 28 - 29 30 - 31 - 32	PLEND# VEW Image: Figure 1 Image: Figure 1 Image: Figure 1 Figure 1 Image: Figure 1 Image: Figure 1 Image: Figure 1 Tune. Ts = 1; Tune. biasc = 0; 2557 Tune. biasc = 0; Tune. lambda = 0.(2505 Tune. lambda = 0.(Tune. objOpt = 1; 2405 Tune. uLL = -0.5* Tune. uLL = 245.* 2405	Figure 1 ever transf Tools Devices Window Help b b C C C C C C C C C C C C C C C C C C	Reference trajectory System Response	_ 8 ×
23 - 24 - 25 - 26 - 27 - 28 - 29 30 - 31 - 32 33	PLEND# VEW Image: Fill of the state of	Figure 1 ever froat Tools Destore Window Help tools Destore Window Help Performance of R 40 41 42 43	Reference trajectory System Response	_ 8 ×
23- 24- 25- 26- 27- 28- 29 30- 31- 32 33 34	PLEND# VEW Image: Comment of the second of the secon	Figure 1 ever front Tools Detator, Window Help Performance of Performance of A 40 41 42 43 Time	Reference trajectory System Response	_ 8 ×
23- 24- 25- 26- 27- 28- 29 30- 31- 32 33 34 35-	PLEND# VEW ■ ■ Graftle ● ■ File Edf VI ■ Graftle ● ● File Edf VI Image: Second	Figure 1 ever front Tools Detatop: Window Help Performance or R 40 41 42 43 Time +1, nCV) ;	Reference trajectory System Response	_ 8 ×
EDITOR 23- 24- 25- 26- 27- 28- 29 30- 31- 32 33 34 35- 36-	PLEND# VEW ■ ■ Graftle ● ■ File Edf VI ■ Graftle ● ● File Edf VI Pile File Edf VI Pile File Edf VI Pile	Figure 1 ever front Tools Detatop: Window Help Performance or R 40 41 42 43 Time +1, nCV) ;	Reference trajectory System Response	_ 8 ×
 ✓ ✓	PLEND# VEW ■ ■ Graftle ● ■ File Edf VI ■ Graftle ● ● File Edf VI Pile File Edf VI Pile File Edf VI Pile	Figure 1 ever tracet Tools Delator, Window Help Performance or Performance or 40 41 42 43 Time +1,nCV); endTime+1)/5));	Reference trajectory System Response	_ 8 ×



We can change this to say 0.3323 where with we will have smaller variation than the previous case. We will have fairly stable response. You can see that this green curve goes to stable and it reaches the setpoint but if we zoom in, we can sorry. If we zoom in, so you can see that the green curve has a deviation from 2.5. Let us close this and we will turn on the bias correction and see how this works. So we have like 0.1 sorry 0.015 as the bias. It is still not reaching the setpoint. So we will have to tune depending on how different the model is.

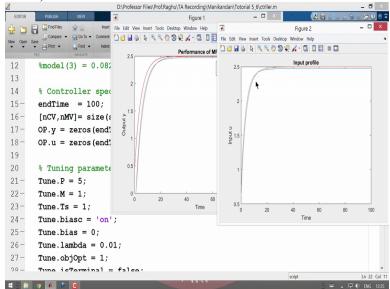
The other way of eliminating this plant model mismatch is to add filters like Kalman filters which estimates the states online using measurements. We can tune the filter to have optimal performance and then we can couple the filter implementation along with MPC so that we will have better control.

(Refer Slide Time 23:02)



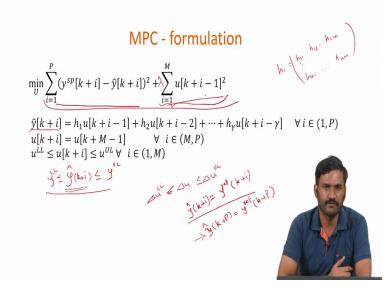
Next next let us at when the reference trajectory is changing how the system will respond? So we will change the reference trajectory as a first order response, first order response with sometime constant as 5 seconds and we will use the same model as the system and see how the system responds. So the idea is we are not restricted to using a single set point. We can give a reference trajectory which is a function of time and we can still control the system fairly, easily. So you can see that the reference trajectory is a first order response and the system was able to reach the trajectory.

(Refer Slide Time 24:08)



We can try changing m which I have put it as 1. Let us put it as 3 so that we will have more room for optimisation. So in that case, the variation will become smoother and easier compared to the case where it is 1. So the orange curve is the new cover which uses one setpoint and the corresponding curve is here. So you can see that it is kind of smooth. We can increase the puff prediction horizon together with control horizon to get a better performance. So in that case, we will sorry let us see here. So in that case we will get easier better approximations and hence we will get better control. Now we have looked at MPC implementation in MATLAB.

(Refer Slide Time 25:45)



We can do other customisations like we can add weights to different elements present in this as well as this. So that will advise the optimiser to put more weightage towards that prediction or less weightage towards that prediction. So we already have a tuning parameter lambda multiplying this. So based on which we we advise the optimiser to weight this the 2nd term such that the first term is either more weight, given more weightage given less weightage.

We can also add other constraints like y of k plus i lying between y of upper limit to y of lower limit, sorry y of lower limit to upper limit. Now this constraint maybe these constraints may not be achievable because we assume the system to be linear and finite impulse response system. So there are plant model mismatches. Because of that, we may not exactly match this constraint but we can give some room for the constraint to be relaxed so that we can still get a feasible solution. So adding these constraint which is on output limits, we will may lead to visibility problems. And we can add other constraints like Delta u being within some Delta u limits and other constraints called coincidence point which will help us reach the set point faster by means of specifying y hat of k plus i to be equal to some y reference of k plus i. But remember, this is also an output constraint. Because of that we may have problem of feasibility. The terminal constraint which is y hat of k plus p equal to y ref of k plus p is easy to implement and it also stabilises the controller. So we may, we will not have oscillatory response with this constraint present.

So MPC formulation is not just restricted to SISO systems, we can always use or extend this to MIMO system. In that case the h1 will become a matrix. So we will have h11 h12 all the way up to h1m and hn1 all the way up to hnn. So similarly we will have for each coefficient like this. So it can be easily extended for MIMO systems. We can as well use nonlinear model to predict y hat of k plus i. But the problem is if we are to use on nonlinear model, the optimisation becomes nonlinear optimisation. So we may end up with local optimum rather than global optimum which is what we get in by solving this optimisation function.

So like the MPC eformulation by itself does not specify which type of model to use. Other than that, if we were to if we have to solve optimisation problem in accorded a programming framework or using linear programming principles, the model has to be linear. Other than that, there is no specification of how, which type of model to use for prediction. So we can use any type of model for prediction. We can add constraint on both us, ys, Delta us and other such constraints which may occur and we can as well customised the objective function based on minimizing u square or minimising Delta u square, minimising time, et cetera.

So the MPC framework by itself is flexible enough but it has to be customised and each constraint we add may each equality constraint we we add and its output constraint we add may make the optimisation infeasible. So the stability and feasibility of this MPC framework has to be theoretically proved. So like this, we can implement MPC in MATLAB. There is also a toolbox called MPC Toolbox in MATLAB which is very easy to learn. At the same time, it will be it will have more diagnostic facilities and other options.

So I have shown here the traditional way of coding the MPC by hand and implementing it for any system. So with this, I finish MPC tutorial.