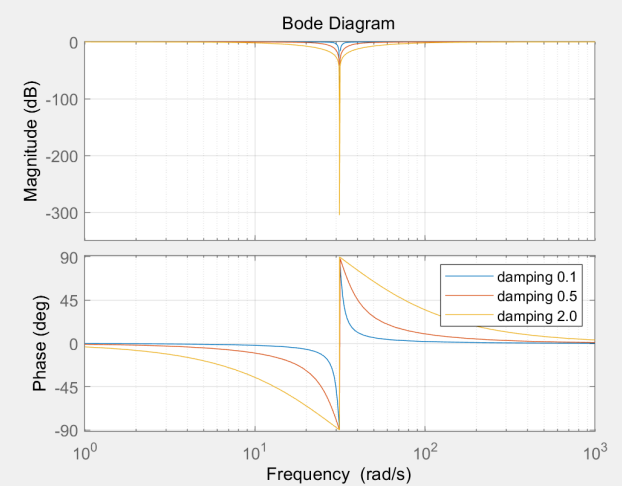
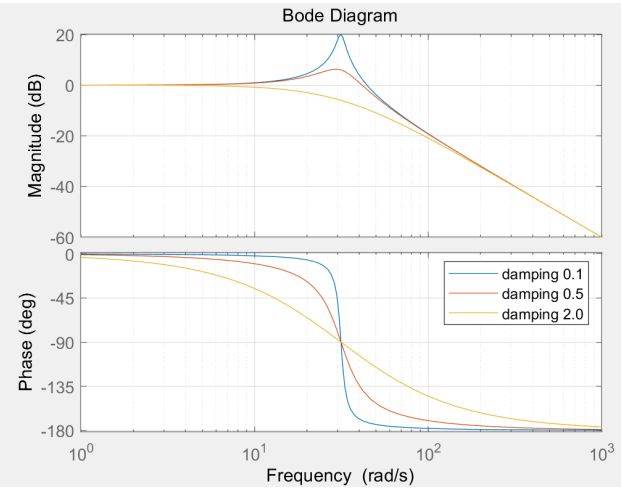
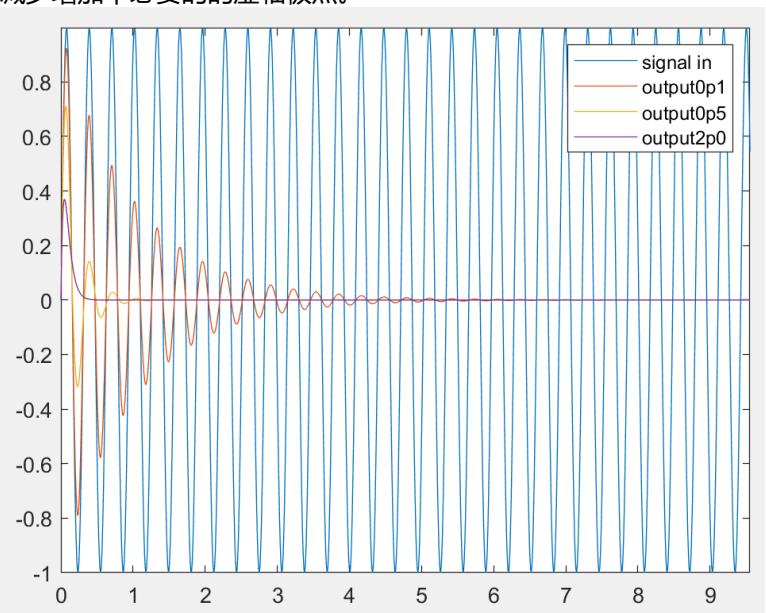
20190328---基于C语言的数字滤波器实现

笔记本： gxm_blaze的笔记本
创建时间： 2019/3/28 0:19 **更新时间：** 2019/5/28 13:56
作者： 晓明
URL： https://baike.baidu.com/item/%E4%BA%8C%E9%98%B6%E5%B8%A6%E9%80%...

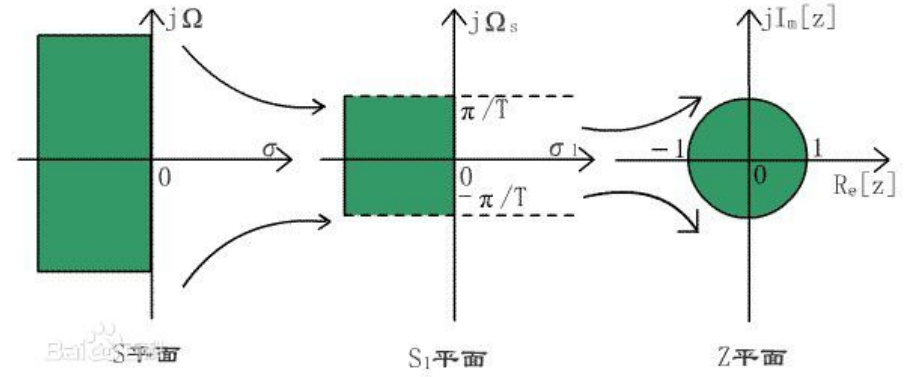
1. 传统滤波器分为低通，高通，带通，带阻四种滤波器，他们有统一的S函数形态
2. $G(s)=\frac{p_2*s^2+p_1*s+p_0}{s^2+e_1*s+e_0}$，或者以角频率表示，$G(s)=\frac{p_2*s^2+p_1*s+p_0}{s^2+\frac{\omega_0}{Q}*s+\omega_0^2}$，
3. 上面的式子中，p2和p1决定滤波器类型
4. 双二阶低通滤波器为：$G(s)=\frac{p_0}{s^2+\frac{\omega_0}{Q}*s+\omega_0^2}$，其衰减为
$$A(\Omega)=G(j*\omega)*G(-j*\omega)=\frac{Q^2}{(\Omega^4*Q^2-2*\Omega^2*Q^2+\Omega^2+Q^2)}$$
5. 双二阶高通滤波器为：$G(s)=\frac{s^2}{s^2+\frac{\omega_0}{Q}*s+\omega_0^2}$，其衰减为
$$A(\Omega)=\frac{-Q^2*\Omega^4}{(\Omega^4*Q^2-2*\Omega^2*Q^2+\Omega^2+Q^2)}$$
6. Dotx上代码的实现上，也是基本如此，统一的分母形式，改变p2，p1可以得到不同类型的滤波器
    1. Notch Filter： `filter_setTf_sca( filt, R_(1.0), R_(0.0), freq*freq, R_(1.0), damp*freq, freq*freq, Ts, freq )`
    2. Notch Filter: `filter_setTf_sca( filt, R_(1.0), R_(0.0), freq*freq, fac, damp*freq, freq*freq, Ts, freq )`
    3. LowPass Filter: `filter_setTf_sca( filt, R_(0.0), R_(0.0), freq*freq, R_(1.0), damp*freq, freq*freq, Ts, FILTER_NOPREWRAP )`
    4. HighPass Filter： `filter_setTf_sca( filt, R_(1.0), R_(0.0), R_(0.0), R_(1.0), damp*freq, freq*freq, Ts, FILTER_NOPREWRAP )`
7. 公式中的R就是1/Q，也可直接叫做阻尼系数，相同的中心频率下，R表示的是带宽或者通带升降的速度

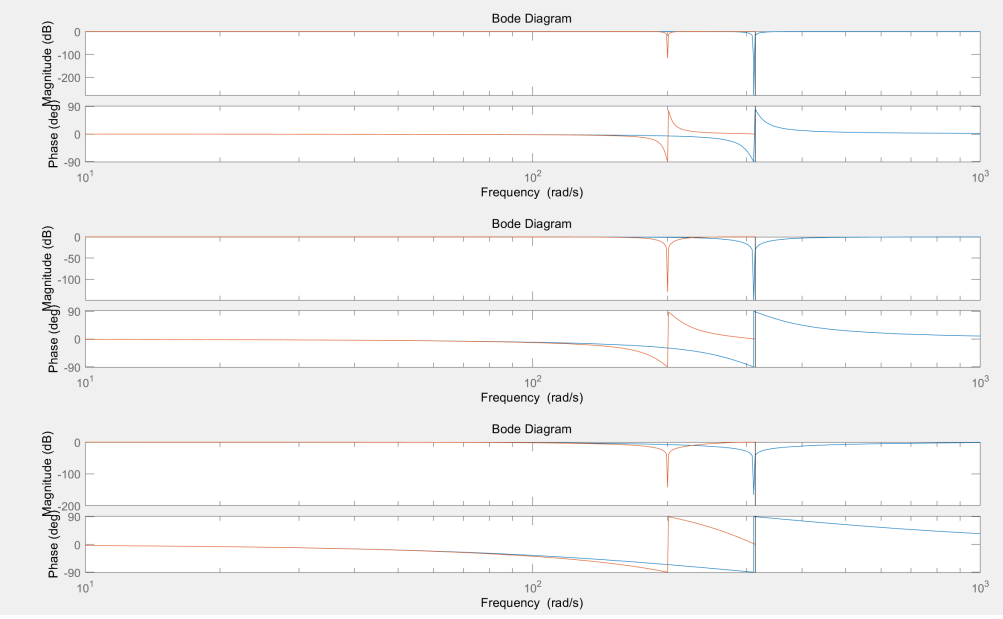
8. 



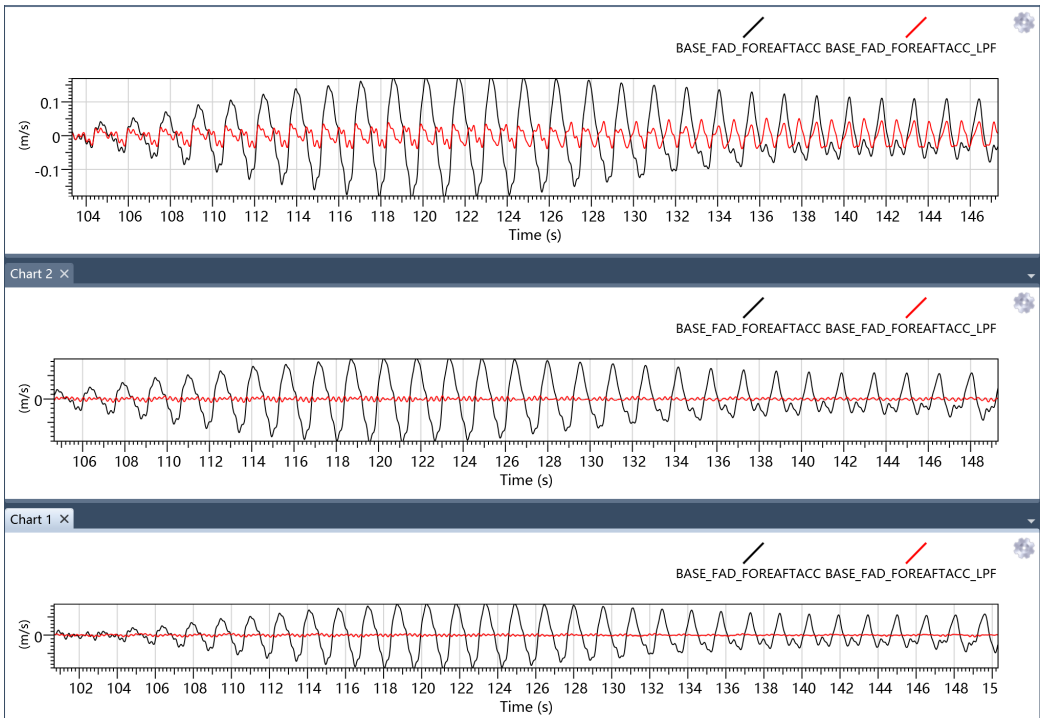
9. 从时间尺度上来看，阻尼系数R的作用就是当信号输入发生阶跃后，滤波器自生需要稳定的时间。低的阻尼系数可以带来较为窄的选择性，但是，需要信号自身足够稳定和缓变才可以使用，个人感觉，对于大部分控制用途的滤波器，阻尼系数应该尽可能设定在1.0以上，减少增加不必要的的虚轴极点。
10. 

11. 因此对这些二阶滤波器进行S域到Z域的变换就可以得到控制器所需的离散滤波器
12. Z到S用的是一次展开，$$z = e^{sT} = \frac{e^{sT/2}}{e^{-sT/2}} \approx \frac{1+sT/2}{1-sT/2}$$

13. S到Z用的是线性映射 $\Omega = \frac{2}{T}\tan(\frac{\omega}{2})$ ， $\omega = 2\arctan(\frac{\Omega T}{2})$ ，采用欧拉方程cosx=(e^jx+e^-jx)/2，sinx=(e^jx-e^-jx)/2j
14. 将这个变换由虚轴延伸到整个复平面 $s = c \bullet th\left(\frac{s_1 T}{2}\right) = c\frac{1 - e^{-s_1 T}}{1 + e^{-s_1 T}}$ ，
15. 考虑使用欧拉方程， $sinh(x) = \frac{e^x - e^{-x}}{2}$ ， $cosh(x) = \frac{e^x + e^{-x}}{2}$ ，
$tanh(x) = \frac{sinhx}{coshx}$， $\cosh^2 x - \sinh^2 x = 1$
16. 最终可以得到S到Z的变换
17. $s = \frac{2}{T}\frac{1 - z^{-1}}{1 + z^{-1}}$
18. 

19. 因为S到Z变换是一个非线性变换，所以，S域的滤波器为了得到准确的滤波频率必须要做适当的预畸变，freq=2/sampling_T*tan(target_freq*sampling_T/2)，这种情况下Z变换后的滤波器频率会正好落在目标频率上。以10ms采样的系统为例，系统最大能设计的滤波器带宽为314rad，在离散域设计一个200rad的带阻滤波器，需要准备的连续域滤波器带宽在300rad左右，双线性变换才能刚好满足要求。
20. 

21. 最后，作为在Dot X上的实列，采用Damping系数R=1.0，过滤3p，过滤3，6，9p，过滤，3，6，9，12p之后的波形如下，可以看到Dot X自带的滤波器工作非常正常。（关闭FADamping输出情况下）

22. 

23. 无关项：对于输入u输出y表达的差分方程，可以表达为积分累加增益网络，其中对于输入u的部分，和输出y的部分，可以完全解耦，u的闭环决定了x为状态量额传递函数，多个复合x不同阶次的输入，可以用线性的方式，定义u的不同阶次导数累加实现。

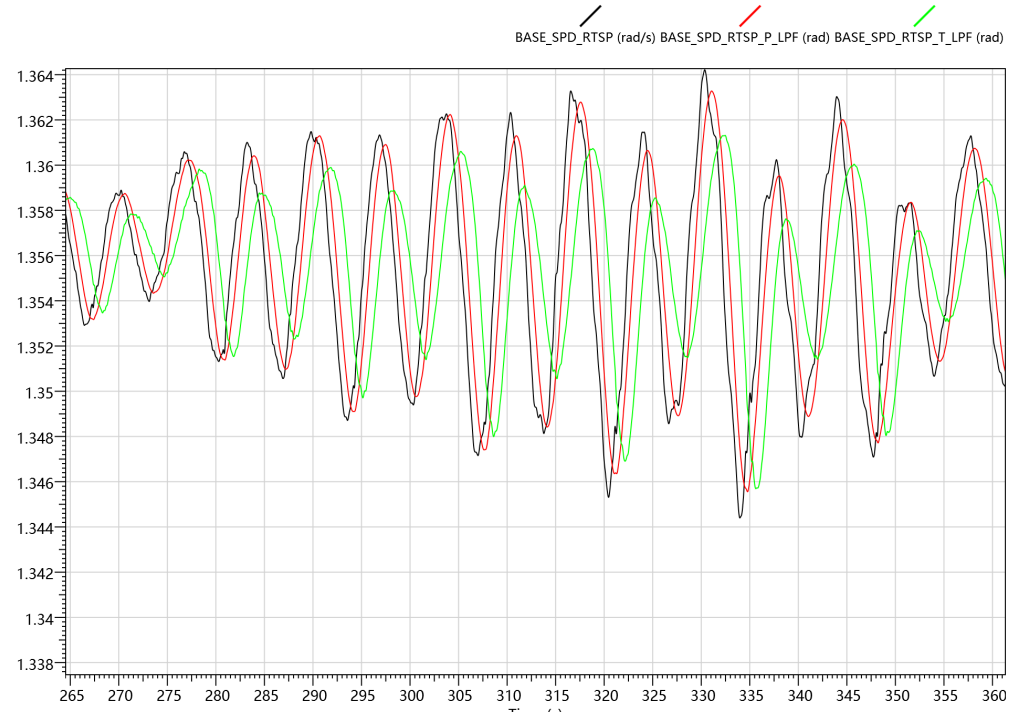
20190503

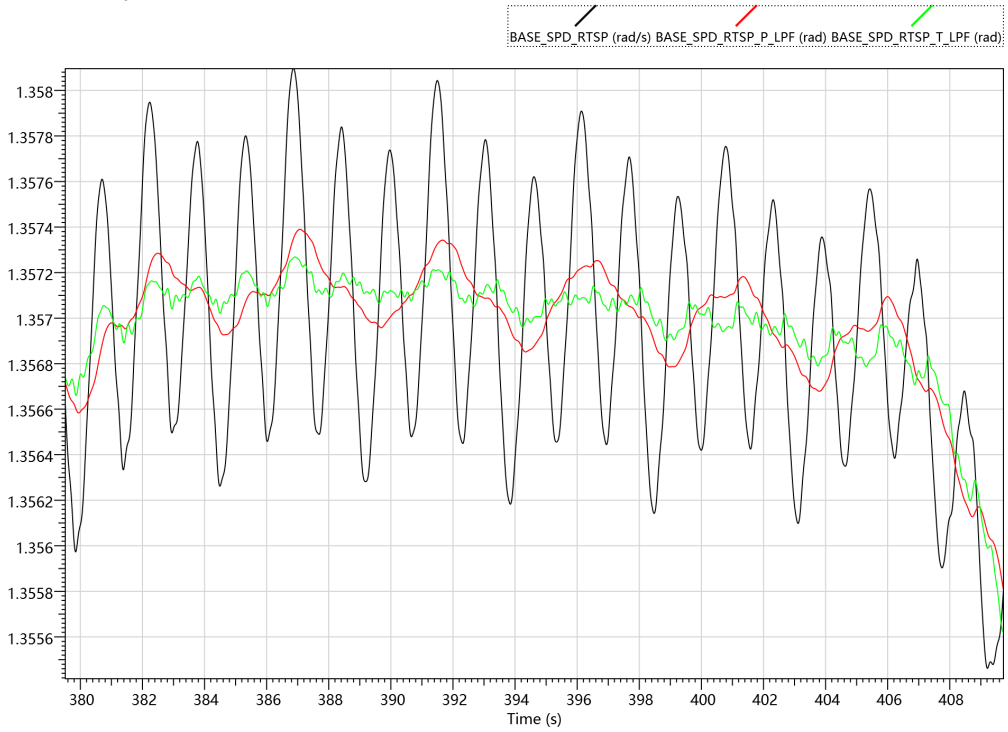
1. 修正对于Damping系数的看法，如同Bode图展示的，Damping系数还会带来频率点附近的相移，Damping系数取大了之后，相移区域会扩大，如果在这个附近存在扰动，则会引发控制器的震荡。
2. 如下RotSpd的Pitch滤波器和Torque滤波器仅差了一个1P的滤波器，但是可以看到，Damping系数太大了以后，出现了一个1p附近的振荡，而且Torque滤波器的相位比Pitch滤波器延迟了大约60°，比原始信号延迟了30°

```
* ------------------------------------------------------------------------------------------------ *
* Rotor Speed Controller                                                                           *
* ------------------------------------------------------------------------------------------------ *
* Filters on rotor speed for pitch controller *
* D  1 -  RotSp_Pit_NF1P         :  1.0          [-]          Notch at 1P, damping factor             *
D  1 -  RotSp_Pit_NF3P         :  0.1          * [-]          Notch at 3P, damping factor             *
*D  1 -  RotSp_Pit_NF6P         :  0.05           [-]          Notch at 6P, damping factor             *
*D  1 -  RotSp_Pit_NF9P         :  0.08           [-]          Notch at 9P, damping factor             *
D  2 -  RotSp_Pit_NF1F         :  1.0  10.3 *[-], [rad/s] Notch, damping factor & frequency      *
*D  2 -  RotSp_Pit_NF2F         :  0.2 1.55    [-], [rad/s]  notch filtering, damping factor & frequency    Tower  *
D  2 -  RotSp_Pit_NF3F         :  1.0   3.7 * [-], [rad/s] , damping factor & frequency    Collective Pitch Mode Notch *
* D  2 -  RotSp_Pit_NF4F         :  0.50   4.00  [-], [rad/s] Notch, damping factor & frequency     *
D  2 -  RotSp_Pit_LP1F         :  1.80 10.0      *[-], [rad/s] Low-Pass, damp (:= 2.0) & frequency    *

* Filters on rotor speed for torque controller *
D  1 -  RotSp_Tor_NF1P         :  1.0          [-]          Notch at 1P, damping factor             *
D  1 -  RotSp_Tor_NF3P         :  0.1          *[-]          Notch at 3P, damping factor             *
* D  1 -  RotSp_Tor_NF6P         :  1.0        * [-]          Notch at 6P, damping factor             *
* D  1 -  RotSp_Tor_NF9P         :  1.0        * [-]          Notch at 9P, damping factor             *
D  2 -  RotSp_Tor_NF1F         :  1.0     10.3 *[-], [rad/s] Notch on tower 0.52Hz, damping factor & frequency      *
* D  2 -  RotSp_Tor_NF2F         :  0.05 25.7359  [-], [rad/s] 4.1Hz notch filtering, damping factor & frequency      *
D  2 -  RotSp_Tor_NF3F         :  1.0   3.7  *[-], [rad/s] Notch, damping factor & frequency      *
* D  2 -  RotSp_Tor_NF4F         :  0.50   4.00  [-], [rad/s] Notch, damping factor & frequency      *
D  2 -  RotSp_Tor_LP1F         :  1.80 40.23 * [-], [rad/s] Low-Pass, damp (:= 2.0) & frequency    *
```

3.

4. 

5. 经过修改Torque滤波器的NF1P的Damping系数为0.05后，问题得到了解决，下图中Torque滤波器中的1p信号也得到了有效的清除
6. 

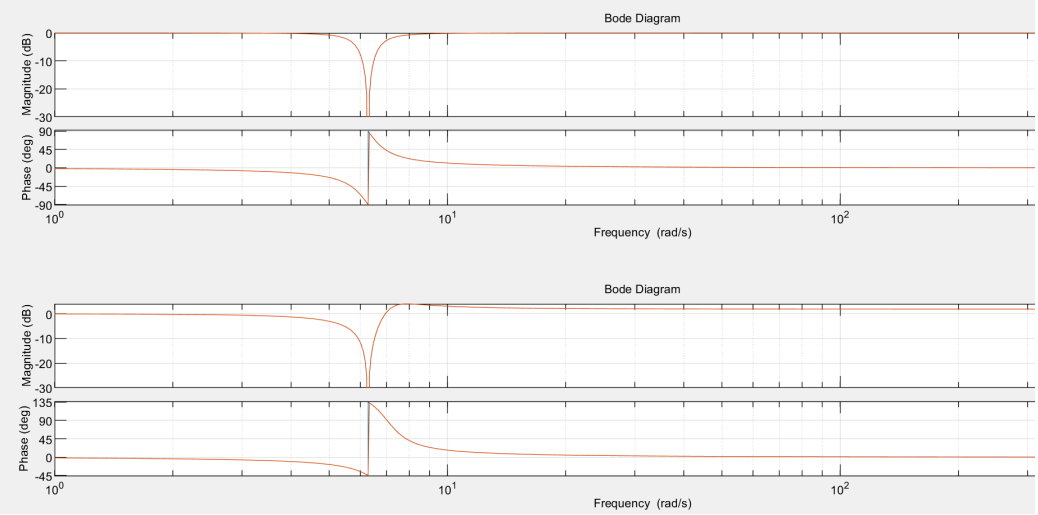
7. 结论：所以Damping系数的选取，应该与信号本身的特质相关，如果信号比较干净，也有比较大的惯性，则取一个较低的Damping有助于后面的控制，如果信号比较糟烂，滤波器的Damping需要大一些，但是后面的控制就很难做，增益系数等都不敢放的太大。
8. 坎贝尔图的分析里，有可能需要吧滤波器的特性放进去，稳定工作区间的+/-10%内不得有滤波器的相位反转区域

20190504

1. 在3米风速的2号Case中发现，在切入转速之前，风轮转速较低的时候，FA Damping的3p动态滤波可能会和塔筒频率（0.52Hz）接近，滤波器的0.5Hz左右信号的输入输出相位出现巨大的反转，导致了系统震荡，切入转速之前需不要避免FA Damping工作。

2. 在5.5米风速1号Case中，在切入转速上，Spd-Torque控制环路配合1p滤波器，仿真过程中现象发生了逐渐的控制失稳，因为仿真系统外部的1p扰动并不存在，所以，存粹是是Spd-Torque控制回路自己的特定频率相位裕度不足造成的，关闭NF1P滤波器问题解决。

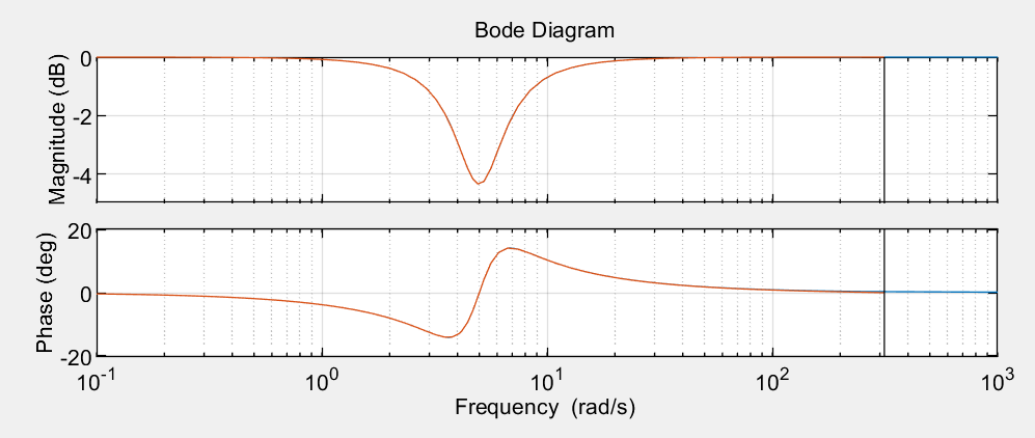
20190520

1. 在DotX中发现一个特殊的Notch滤波器，在常规Notch滤波器上增加了Fac系数，DotX中取值为0.8，通过Matlab仿真可以看出，中心频率的左侧相位裕度明显提升，由延迟90°改为延迟45°，右侧的相位超前增加45°，幅值也由0db变为最大4db。配合低通滤波器的滞后相移，这种Notch滤波器效果应该会比较好，在调稳定苦难时，可以考虑测试这个滤波器。
2.
```
% LOWPASS:  filt_sys = tf([0 freq*damping*0 freq*freq*1], [1    freq*damping, freq*freq]);
% HIGHPASS: filt_sys = tf([1 freq*damping*0 freq*freq*0], [1    freq*damping, freq*freq]);
% NOTCH:    filt_sys = tf([1 freq*damping*0 freq*freq*1], [1    freq*damping, freq*freq]);
% NOTCH2:   filt_sys = tf([1 freq*damping*0 freq*freq*1], [0.8  freq*damping, freq*freq]);
% BANDPASS: filt_sys = tf([0 freq*damping*1 freq*freq*0], [1    freq*damping, freq*freq]);
```
3. 

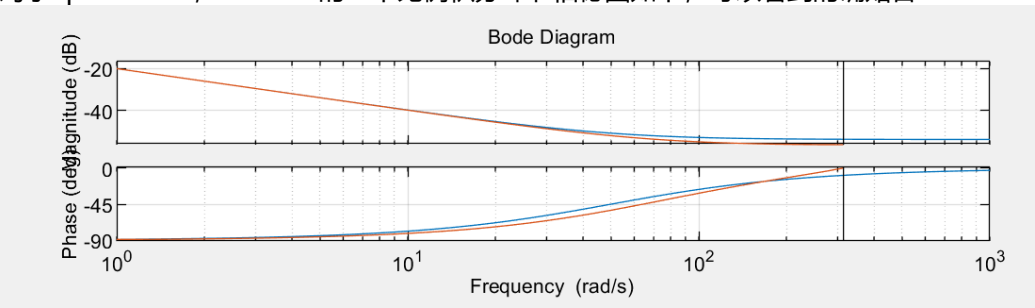
4. 开始在Dot X中的Notch滤波器参数中，添加BandPass滤波器属性，为传动链DTD做准备

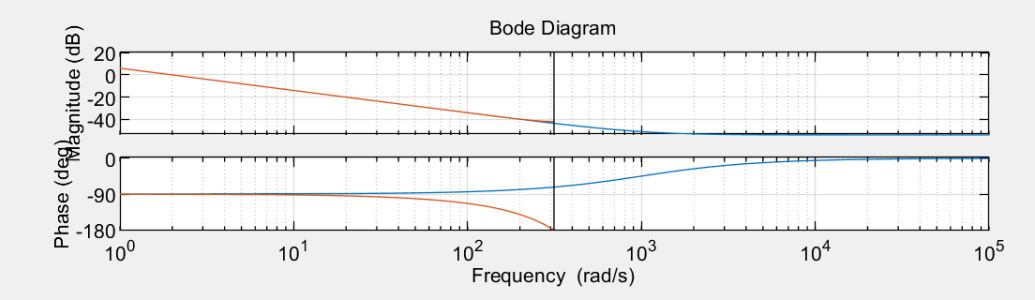
20190522

1. DotX中复用了Notch滤波器结构，补充了BandPass滤波器的功能，需要注意的是，DotX滤波器有串联滤波器的功能，对于BandPass带通滤波器，可能需要考虑并联结构，所以，后续需要花精力做出一个串并联滤波器结构属性和方法来统一操作。
2. 为了提升控制稳定性，在滤波器中补充Lag-Lead类型，放在串联滤波器结构之中，Lead-Lag滤波器如下：
3. $$lead-lag=\frac{(\tau_1 s+1)(\tau_2 s+1)}{(1+\frac{\tau_1}{b}s)(1+b\tau_2 s)}$$
4. 经过简化的后的LeadLag表达式如下，基本保持了和原系统滤波器的一致形态，其中，damping和之前其它滤波器一致，反映的是LeadLag的覆盖相位范围，Mag大于0时为LagLead，小于0时为LeadLag（控制上没有价值）
5.
```
% LOWPASS:  filt_sys = tf([0 freq*damping*0 freq*freq*1], [1    freq*damping, freq*freq]);
% HIGHPASS: filt_sys = tf([1 freq*damping*0 freq*freq*0], [1    freq*damping, freq*freq]);
% NOTCH:    filt_sys = tf([1 freq*damping*0 freq*freq*1], [1    freq*damping, freq*freq]);
% NOTCH2:   filt_sys = tf([1 freq*damping*0 freq*freq*1], [0.8  freq*damping, freq*freq]);
% BANDPASS: filt_sys = tf([0 freq*damping*1 freq*freq*0], [1    freq*damping, freq*freq]);
% LeadLag:  filt_sys = tf([1 damping*freq, freq*freq],   [1    freq*exp(mag)*damping, freq*freq]);
```
6. 选取freq=5，damping=0.5，Mag=0.5的伯德图如下：

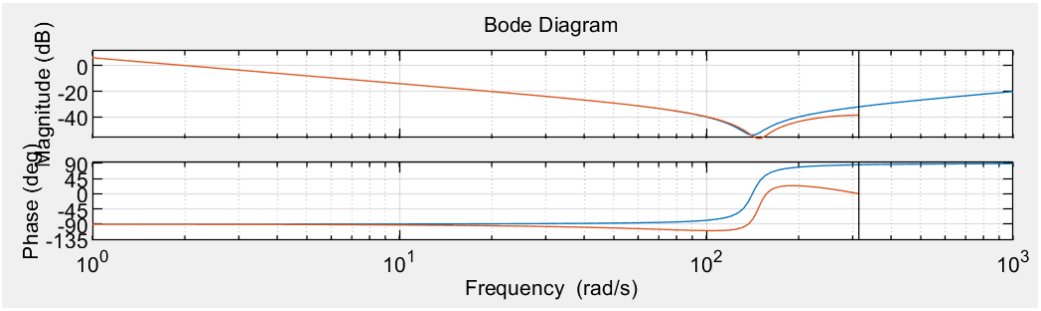
7. 

20190528

1. 新增PID控制的Z变换的Bode图分析，PID控制的S域和Z域在Matlab中的构成如下：
2.
```
% PID:      filt_sys = tf([Ti*Td, Ti, 1],   [0,    Ti, 0]);
% filt0_sys = tf([Kp*Ti*Td, Kp*Ti, Kp],   [0,    Ti, 0]);
% Ki=Kp*sampling_T/Ti;
% Kd=Kp*Td/sampling_T;
% z = tf('z',1);
% z.Ts = sampling_T;
% filt0_z = ((Kp+Kd) + (-Kp+Ki-2*Kd)*z^-1 + Kd*z^-2)/(1-z^-1);
```
3. 对于一个典型的比例积分环节，当Ti大于sampling_T则S域和Z域之间的关系基本贴合
4. 对于kp = 0.002， Ti=0.02的一个比例积分环节伯德图如下，可以看到的确贴合
5. 

6. 但当Ti小于sampling_T，例如对于kp = 0.002， Ti=0.001， Td=0.0的一个接近纯积分环节伯德图如下：
7. 

8. 可以看到模拟PID和数字PID的幅值差别不大，相位差别在30rad/s附近开始分叉，高频如果存在某些高增益，会增加产生谐振的可能性。
9. 对于kp = 0.002， Ti=0.001， Td=0.05的一个接近纯积分环节伯德图如下：

10. 

11. 可以看到经过并联微分环节矫正后，系统的幅值和相位裕度都得到了提升
12. 结论：如果需要使用离散PID，Ti尽量大于采样时间，如果做不到，在考虑各种矫正措施