

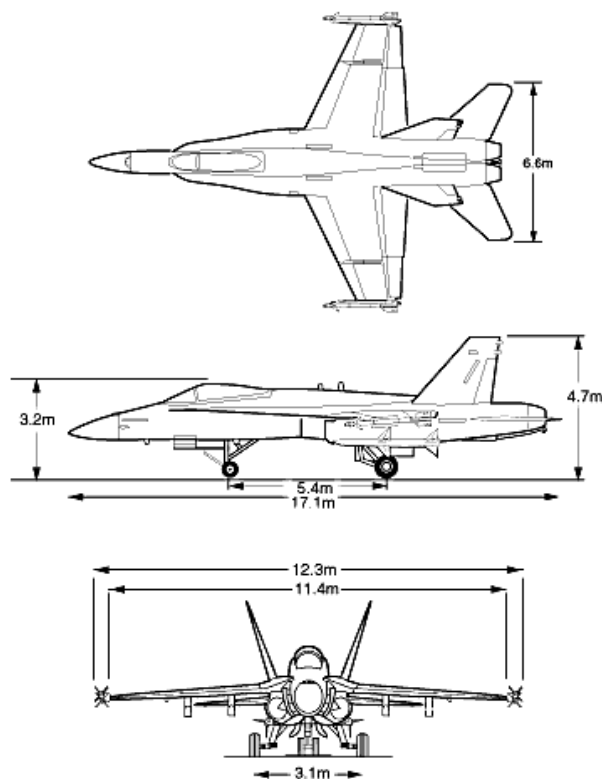
## F-18 HARV Pitch Rate Design Example

Peter Thompson

2-20-02 (updated 4-11-02)

**Introduction:** A special version of the F-18 aircraft was modified by NASA Dryden and called the High Angle-of-Attack Research Vehicle (HARV). In this example Program CC Version 5 is used to design and then analyze a pitch rate stability augmentation system. This example assumes that the reader has some familiarity with control system design. A student can be expected to duplicate a design such as this at the end of his or her first college course in the subject.

A diagram of the standard F-18 is shown below:



**Linear Model:** The aircraft dynamics decouple into longitudinal and lateral-directional axis dynamics. The pitch response is part of the longitudinal axis, together with the forward and vertical directions. (These equations are developed in references such as McRuer D., I. Ashkenas, and D. Graham, Aircraft Dynamics and Automatic Control, Princeton University Press, Princeton, New Jersey, 1973.) A control system design begins with a linear model of the vehicle. The parameters in this model depend on the altitude, speed, and angle-of-attack of the vehicle. The “flight condition” used here is straight and level flight at sea level, with a forward speed of 250 ft/sec and an angle of attack of 30 degrees. The model has the following states, inputs, and outputs:

States:

1.  $u_b$  [ft/sec] = body-axis forward speed
2.  $w_b$  [ft/sec] = body-axis vertical speed
3.  $q$  [rad/sec] = pitch rate
4.  $\theta$  [rad] = pitch attitude

Inputs:

1.  $\delta_e$  [rad] = stabilator angle
2.  $\delta_{v_y}$  [rad] = thrust vector angle
3.  $u_{gb}$  [ft/sec] = horizontal gust, body-axis
4.  $w_{gb}$  [ft/sec] = vertical gust, body-axis
5.  $sw_{gb}$  [ft/sec<sup>2</sup>] = vertical gust, acceleration component

Outputs:

1.  $u_b$  [ft/sec] = body-axis forward speed
2.  $w_b$  [ft/sec] = body-axis vertical speed
3.  $q$  [rad/sec] = pitch rate
4.  $\theta$  [rad] = pitch attitude
5.  $u_s$  [ft/sec] = stability axis forward speed
6.  $\alpha_s$  [rad] = angle-of-attach
7.  $\gamma_s$  [rad] =  $\theta - \alpha_s$  = flight path angle
8.  $\dot{h}$  [ft/sec] = altitude rate
9.  $a_x$  [ft/sec<sup>2</sup>] = forward acceleration at the sensor location, body axis
10.  $a_z$  [ft/sec<sup>2</sup>] = vertical acceleration at the sensor location, body axis

The linear model can be entered using a function call, with coefficients computed or extracted from tables. Here just the numbers are entered:

```
a=[9.500000e-003  0.0257000  -110.00000  -27.860037
-0.13100000  -0.2300000  190.52559  -16.085000
 1.9000000e-003 -3.096697e-003 -0.3100000  0
 0  0  1  0]
b=[-1.2300000  -7.800000e-003 -9.500000e-003 -0.0257000  0
-13.100000  -16.100000  0.1310000  0.2300000  0
-1.2300000  -2.5300000  -1.900000e-003  3.0096697e-003  1.627078e-003
 0  0  0  0  0]
c=[1  0  0  0  0
 0  1  0  0  0
 0  0  1  0  0
 0  0  0  1  0
 0.8660254  0.5000000  0  0  0
-2.272727e-003  3.936479e-003  0  0  0
 2.272727e-003 -3.936479e-003  0  0  1
 0.5000000  -0.8660254  0  0  220
 9.500000e-003  0.0257000  0  0  0
-0.1437300  -0.2092521  2.0770000  0  0]
d=[0  0  0  0  0
 0  0  0  0  0
 0  0  0  0  0
 0  0  0  0  0
 0  0  0  0  0
 0  0  0  0  0
 0  0  0  0  0
 0  0  0  0  0
-1.2300000  -7.800000e-003 -9.500000e-003 -0.0257000  0
-4.8590000  0.8510000  0.1437300  0.2092521  -0.0109014]
p=pack(a,b,c,d)
```

These commands can be copied and then pasted at the Program CC command prompt. The last line, with the “pack” function, places the matrices into a variable called a “quadruple,” also called a state space system. The open loop response of the vehicle is characterized by the “poles” of the system; complex number that are the eigenvalues of the “a” matrix:

```
CC>poles(p)
ans =
-0.0421760 + 0.1707075j
-0.0421760 - 0.1707075j
-0.2230740 + 0.8661532j
-0.2230740 - 0.8661532j
```

**Open Loop Frequency Response:** Classical control design typically starts with transfer function models of the system. The transfer function from the stabilator to the pitch attitude response is computed from the quadruple and then displayed:

```
CC>qde=gepper(p(3,1))
CC>qde

qde(s) = 
$$\frac{-1.23s(s+0.01653)(s+0.1729)}{(s^2 + 0.08435s + 0.03092)(s^2 + 0.4461s + 0.8)}$$

```

This easy transfer back and forth between modern (matrix-based) and classical control (transfer function based) models is an excellent feature of Program CC. Another way to display the transfer function used the “sho” command, short for “shorthand:”

```
CC>sho(qde)

qde(s) = 
$$\frac{-1.23(0)(0.01653)(0.1729)}{[0.2399, 0.1758][0.2494, 0.8944]}$$

```

The numbers in parentheses are the real poles and zeros, and the numbers in the square brackets are the damping ratios and natural frequencies of complex poles and zeros. In this example the “phugoid” mode has a natural frequency of 0.1758 rad/sec, and the “short period” mode has a natural frequency of 0.8944 rad/sec. The partial fraction expansion is used to help understand system properties, which is demonstrated below:

```
CC>pfe(qde)

qde(s) = 
$$\frac{0.05853s+0.006047}{[(s+0.04218)^2+0.1707^2]} - \frac{1.289s+0.1565}{[(s+0.2231)^2+0.8662^2]}$$

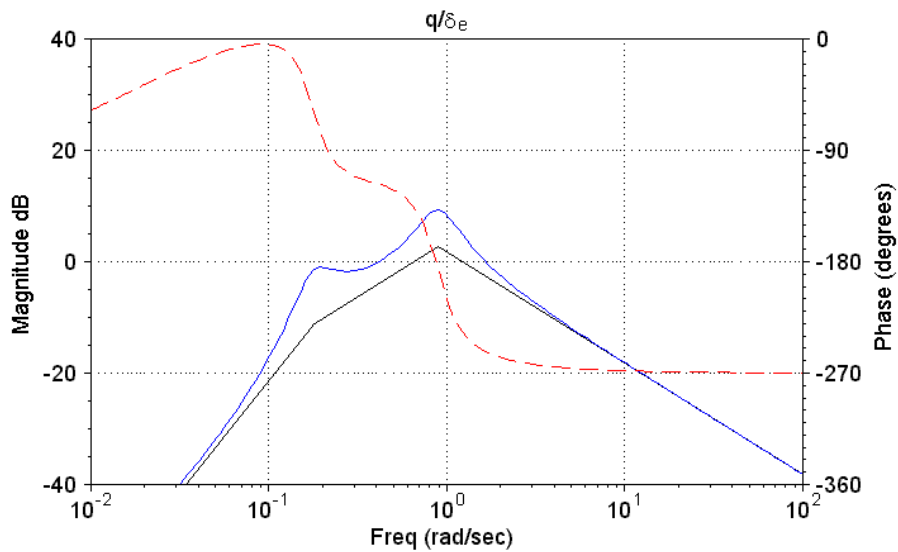
```

More enlightening is the frequency response of this system, created with the following command:

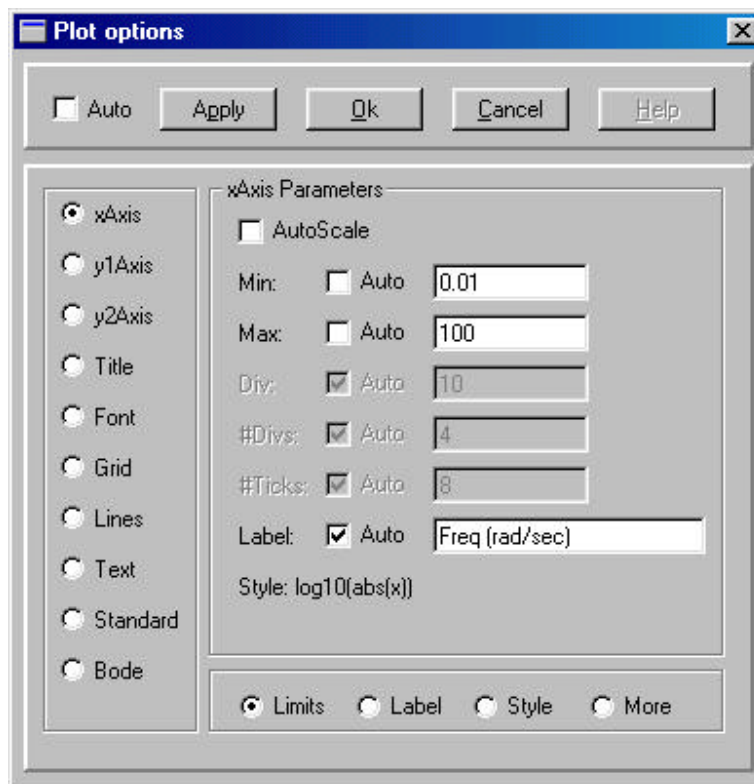
```
CC>bode(qde)
```

The following variation of this command uses standard axis limits, includes magnitude asymptotes, includes a title, and increases the fontsize:

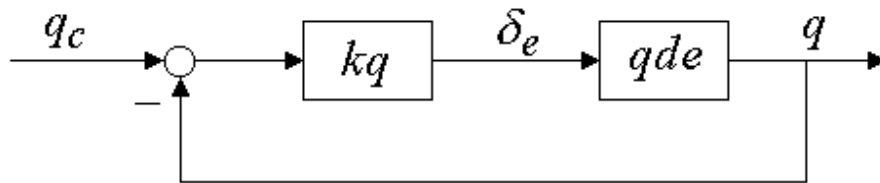
```
CC>bode(qde,'std','asy','title','q/\delta_e','fontsize',14)
```



The blue line is the magnitude of the frequency response, and the red dashed line the phase. The sharp breaks in the black magnitude asymptote correspond to the open loop poles and zeros. Changes to the plot can be made using the “plot option” dialog box, called by double-clicking on the plot:



**Pitch Rate Stability Augmentation:** The F-18 was one of the first operational aircraft designed with relaxed static stability. The aerodynamic stability was reduced to enhance performance and the flight control system was designed to augment the stability and meet the military flying qualities requirements. Augmentation such as this is now standard on military and commercial aircraft, but not so on general aviation aircraft. A block diagram of a pitch rate control system is shown below:



The signals are  $q$  = pitch rate,  $q_c$  = pitch rate command, and  $\delta_e$  = elevator. The system to be controlled is  $qde$ , and the controller is the transfer function  $kq$ , soon to be selected. With this control system in place the pilot's stick input generates a pitch rate command, and the "effective vehicle," the vehicle as seen by the pilot, is the closed loop response of this system. Block diagrams such as this are not part of Program CC. The closed loop response is computed using algebra.

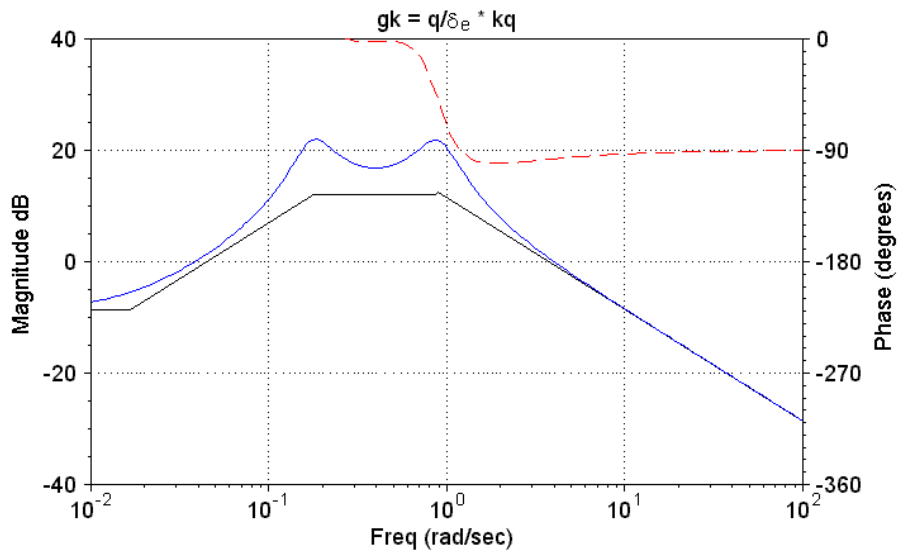
A common controller for this type of problem is called the "superaugmented pitch rate controller." The basic idea is to use an integrator to increase the gain of the low frequency response, include a zero at or near the short period frequency, and then adjust the gain to give the desired speed of response. Here the zero is placed at 0.86 rad/sec and a bandwidth of 4 rad/sec is chosen. This bandwidth is consistent with high performance aircraft. The controller can be designed algebraically:

```
CC>kq=(s+.86)/s; gk=qde*kq; gk=-gk/abs(gk(j*4)); kq=gk/qde
CC>kq
```

$$kq(s) = \frac{-3.033(s+0.86)}{s}$$

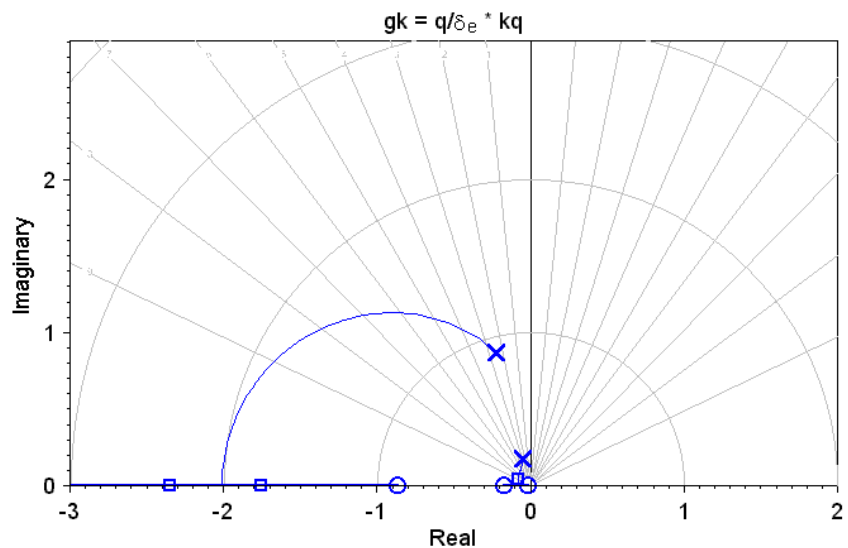
The supraugmented pitch rate controller is a proportional-integral controller. The significant thing about the supraugmented pitch loop, or any other SCAS architecture applied to an airframe, is that it is high gain, high authority flight control systems that changes the airframe dynamics much more than earlier generations of aircraft flight control systems. This is made possible by the introduction of highly redundant digital flight control systems in pioneering designs like the F-18 and the Space Shuttle. The product of the system and controller,  $gk$ , is called the "loop transfer function" of the system. Its frequency response is used to infer stability and closed loop response of the system. A bode plot of  $gk$  is shown below:

```
bode(gk,'std','asy','title','gk = q/\delta_e * kq','fontsize',14)
```



The magnitude response crosses the 0 dB line at 4 rad/sec, according to the design, and the phase at this frequency is just block  $-90$  degrees, indicated a stable and robust design. The actuator and sensor dynamics, plus digital control system lags, would subtract from this phase, but will not significantly affect this design. The root locus is used to track the closed loop poles as the gain of the system changes, and is plotted with the following command:

```
rl(gk,'zeta','xlim',-3,2,'title','gk = q/\delta_e * kq','clgain',1,'fontsize',14)
```



The command line has several options; and the simpler command `rl(gk)` would generate a quite usable plot. Here we have specified the x-axis limits, put magnitude and zeta lines on the background, and included the closed loop poles as boxes at unity gain. All of this can be done using the “plot option” dialog box.

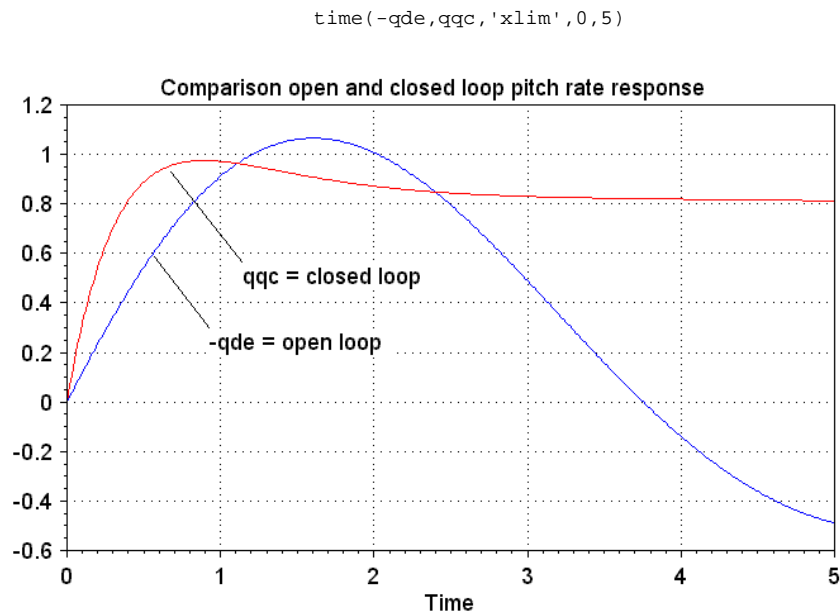
**Closed loop response:** The closed loop transfer function is computed using the “feedback” operator (the vertical bar in the command line below), and then displayed two different ways:

```
CC>qqc=(qde*kq)|1; qqc; sho(qqc)
```

$$qqc(s) = \frac{3.731(s+0.01653)(s+0.1729)(s+0.86)}{s^4 + 4.261s^3 + 4.784s^2 + 0.6997s + 0.0339}$$

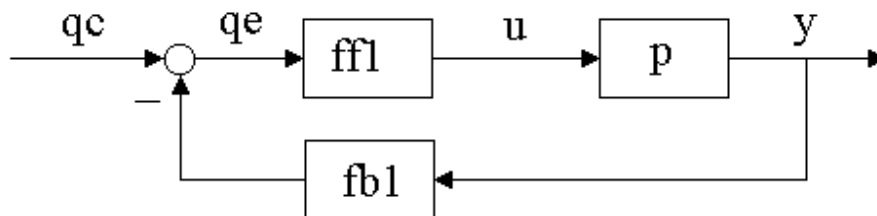
$$qqc(s) = \frac{3.731(0.01653)(0.1729)(0.86)}{[0.8917, 0.09081](1.75)(2.349)}$$

A comparison of the open and closed loop pitch rate responses are shown in the next figure:



The function call defaults to a unit step response, and specifies the first 5 seconds. The title and annotation were added to the plot using the plot dialog box. In the open loop response the pilot is directly controlling the elevator, and the drop-off due to the phugoid response is dramatic after 2 seconds. This is not necessarily bad, since the pitch rate command is rarely used for this long, and the attitude adjustment is usually completed in one or two seconds. The closed loop response, however, is more consistent and predictable, and results in a lower workload for the pilot, who has enough other tasks to perform.

**Stripchart:** All of the other outputs will respond to a pitch rate command, and the stripchart is a good way to display these responses. The same pitch rate controller is applied to the state space model of the open loop system. A block diagram that includes the state space system is shown below:



The fb1 and ff1 matrices are used to connect the third output (the pitch rate) to the first input (the elevator):

```
CC>fb1=(0,0,1,0,0,0,0,0,0,0)
CC>ff1=(kq;0;0;0;0)
CC>c11=(p*ff1)|fb1
```

The feedback operator is the vertical bar, which contains the forward path, (p\*ff1) before the operator, and the feedback path, fb1, after the operator. To check consistency with the transfer function closed loop system:

```
CC>sho(gepper(c11(3,1)))
ans(s) = 
$$\frac{3.731(0.01653)(0.1729)(0.86)}{[0.8917, 0.09081](1.75)(2.349)}$$

```

We should get the same closed loop transfer function, and we do. Compute a closed loop simulation to a unit step response in the pitch rate command:

```
CC>(yt,y)=sim(c11,'tmax',5)
```

The matrix y has 10 columns corresponding to the 10 outputs. The times are in the vector yt. The parameter tmax specifies 5 seconds, and the command defaults to a unit step with 500 points. To plot the result:

```
stripchart(yt,y,'title','Response to unit step qc',...
'ylabel','u_b,w_b,q,\theta,u_s,\alpha,\gamma,hdot,ax,az')
```

The ylabel plot option is used to label the outputs, with the names separated by commas. The backslashes are used to specify Greek letters. Different units can be used for the inputs and outputs, changing for example radians to degrees. The resulting stripchart is shown to finish this example:

