```
% Uncertainty
% Here you are seting the standard deviation of the uncertianty
% Values next to each setting are useful for understanding the scale
% The values for random are one standard deviation of random error.
% The falues for bias are one standard deviations offset.
% k,m,b
param.uncertian_param = {'m(2)','Jc','d','mu'};
param.D_in_param.random  = 0.2.*[param.m(2),param.Jc,param.d,param.mu];
param.D_in_param.bias    = 0.0.*[param.m(2),param.Jc,param.d,param.mu];
% F
param.uncertian_u = [false,false];
param.D_in_u.random      = [0.0,0.0];
param.D_in_u.bias        = [1.0*sum(param.m),0.0];
% z,z_dot
param.uncertian_x = [false,false,false,false,false,false];
param.D_out.random       = [0,0,0,0,0,0];
param.D_out.bias         = [0,0,0,0,1.0,0];
% z,z_dot - measured
param.uncertain_N = [false,false,false,false,false,false];
param.N.random           = [0,0,0,0,0,0];
param.N.bias             = [0,0,0,0,0,0];
```

```
      function output = uncertainty(self,input,uncertainty,keys)

            scale =
uncertainty.random.*(rand(1,length(uncertainty.random)).*2-1);

            if isstruct(input)
                output = input;
                for i = 1:length(keys)
                    variable = keys{i};
                    code = ['output.',variable,' = input.',variable,'+
scale(i) + uncertainty.bias(i);'];
                    eval(code)
%                     disp(variable)
%                     eval(['disp(input.',variable,')'])
%                     eval(['disp(output.',variable,')'])
                end
            else
                output = zeros(size(input));
                for i = 1:length(input)
                    if keys(i)
                        output(i) = input(i) + scale(i) +
uncertainty.bias(i);
                    else
                        output(i) = input(i);
                    end
                end
            end

        end
```