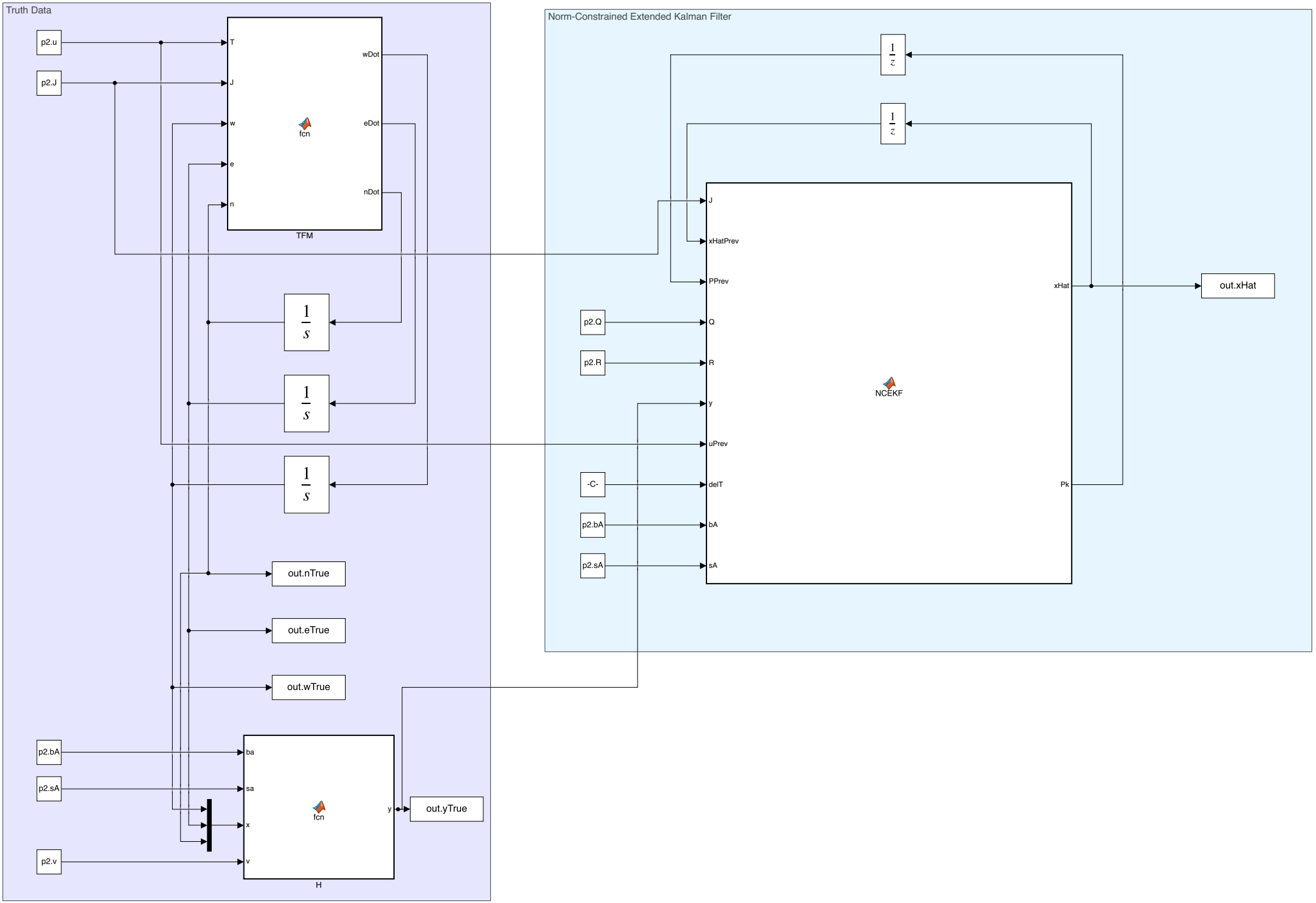
Truth Data
p2.u
p2.J
T
J
w
e
n
fcn
wDot
eDot
nDot
TFM
1/s
1/s
1/s
out.nTrue
out.eTrue
out.wTrue
p2.bA
p2.sA
p2.v
ba
sa
x
v
fcn
y
H
out.yTrue
Norm-Constrained Extended Kalman Filter
1/z
1/z
J
xHatPrev
PPrev
p2.Q
p2.R
y
uPrev
-C-
delT
p2.bA
p2.sA
bA
sA
NCEKF
xHat
Pk
out.xHat

```
function y = fcn(ba, sa, x, v)

    function wx = skewSymmetric(w)
        wx = [0, -1*w(3), w(2);
             w(3), 0, -1*w(1);
             -1*w(2), w(1), 0];
    end


    function Yaq = Y_aa_q(a, e, n)

        Ya = [n*eye(3)-skewSymmetric(e), -e];
        Yb = [skewSymmetric(a), a; -a', 0];
        Yaq = Ya*Yb;

    end

e = x(4:6);
n = x(7);

y = [zeros(3,3), Y_aa_q(ba, e, n);
     zeros(3,3), Y_aa_q(sa, e, n)] * x + v;

end
```

```
function [xHat, Pk] = NCEKF(J, xHatPrev, PPrev, Q, R, y, uPrev, delT, bA, sA)

    function wx = skewSymmetric(w)
        wx = [0, -1*w(3), w(2);
             w(3), 0, -1*w(1);
             -1*w(2), w(1), 0];
    end

    function Yaq = Y_aa_q(a, e, n)

        Ya = [n*eye(3)-skewSymmetric(e), -e];
        Yb = [skewSymmetric(a), a; -a', 0];
        Yaq = Ya*Yb;

    end

    function dstate = TFM(state, T, J)

        w = state(1:3);
        e = state(4:6);
        n = state(7);

        wDot = J\(T-skewSymmetric(w)*J*w);

        eDot = 0.5*(n*eye(3) + skewSymmetric(e))*w;
        nDot = -0.5*e'*w;

        dstate = [wDot;eDot;nDot];
    end

% predictor
wHatPrev = [xHatPrev(1);xHatPrev(2);xHatPrev(3)];
eHatPrev = [xHatPrev(4);xHatPrev(5);xHatPrev(6)];
nHatPrev = xHatPrev(7);

rowA = [J\(-skewSymmetric(wHatPrev)*J + skewSymmetric(J*wHatPrev)), zeros(3,3), zeros(3,1)];
rowB = [0.5*(nHatPrev*eye(3) + skewSymmetric(eHatPrev)), -0.5*skewSymmetric(wHatPrev), 0.5*wHatPrev];
rowC = [-0.5*eHatPrev', -0.5*wHatPrev', 0];

FPrev = eye(7) + delT * [rowA;rowB;rowC];
LPrev = [delT*inv(J);zeros(3,3);zeros(1,3)];

f_hat_prev = TFM([wHatPrev;eHatPrev;nHatPrev], 0, J);

xHatMinus = xHatPrev + f_hat_prev*delT;
wHatMinus = xHatMinus(1:3);
eHatMinus = xHatMinus(4:6);
nHatMinus = xHatMinus(7);

PkMinus = FPrev*PPrev*FPrev' + LPrev*Q*LPrev';

PkMinus_1 = PkMinus(:,1:3);
PkMinus_2 = PkMinus(:,4:7);

PkMinus_ww = PkMinus(1:3,1:3);
PkMinus_qw = PkMinus(4:7,1:3);
PkMinus_wq = PkMinus(1:3,4:7);
PkMinus_qq = PkMinus(4:7,4:7);
```

```
% corrector
M = eye(6);
H = [zeros(3,3), Y_aa_q(bA, eHatMinus, nHatMinus);
       zeros(3,3), Y_aa_q(sA, eHatMinus, nHatMinus)];

W = H*PkMinus*H' + M*R*M';
K_w = PkMinus_1'*H'*inv(W);
h_y = H*xHatMinus;
wHat = wHatMinus + K_w*(y - h_y);

K_tilde = PkMinus_2'*H'*inv(W);
r_k_vec = y - h_y;
r_k_scl = r_k_vec'*inv(W)*r_k_vec;

q_tilde = [eHatMinus;nHatMinus] + K_tilde*r_k_vec;

K_qk = K_tilde + 1/r_k_scl * (1/norm(q_tilde) - 1)*q_tilde*r_k_vec'*inv(W);

qHat = [eHatMinus;nHatMinus] + K_qk*r_k_vec;

xHat = [wHat;qHat];

Pk_ww = PkMinus_ww - K_w*H*PkMinus_1 - PkMinus_1'*H'*K_w' + K_w*W*K_w';
Pk_wq = PkMinus_wq - K_w*H*PkMinus_2 - PkMinus_1'*H'*K_qk' + K_w*W*K_qk';
Pk_qw = PkMinus_qw - K_qk*H*PkMinus_1 - PkMinus_2'*H'*K_w' + K_qk*W*K_w';
Pk_qq = PkMinus_qq - K_qk*H*PkMinus_2 - PkMinus_2'*H'*K_qk' + K_qk*W*K_qk';

Pk = [Pk_ww, Pk_wq; Pk_qw, Pk_qq];

end
```

```
function [wDot, eDot, nDot] = fcn(T, J, w, e, n)

    function wx = skewSymmetric(w)
        wx = [0, -1*w(3), w(2);
              w(3), 0, -1*w(1);
              -1*w(2), w(1), 0];
    end

wDot = J\(T-skewSymmetric(w)*J*w);

eDot = 0.5*(n*eye(3) + skewSymmetric(e))*w;
nDot = -0.5*e'*w;

end
```