```
function [x,P] = ekf(fstate,x,P,hmeas,z,Q,R)

    [x1,A]=jaccsd(fstate,x);    %nonlinear update and linearization at current state
    P=A*P*A'+Q;                 %partial update
    [z1,H]=jaccsd(hmeas,x1);    %nonlinear measurement and linearization
    P12=P*H';                   %cross covariance
    % K=P12*inv(H*P12+R);       %Kalman filter gain
    % x=x1+K*(z-z1);            %state estimate
    % P=P-K*P12';               %state covariance matrix
    R=chol(H*P12+R);            %Cholesky factorization
    U=P12/R;                    %K=U/R'; Faster because of back substitution
    x=x1+U*(R'\(z-z1));         %Back substitution to get state update
    P=P-U*U';                   %Covariance update, U*U'=P12/R/R'*P12'=K*P12.

end

function [z,A]=jaccsd(fun,x)

    % JACCSD Jacobian through complex step differentiation
    % [z J] = jaccsd(f,x)
    % z = f(x)
    % J = f'(x)
    %
    z=fun(x);
    n=numel(x);
    m=numel(z);
    A=zeros(m,n);
    h=n*eps;
    for k=1:n
        x1=x;
        x1(k)=x1(k)+h*i;
        A(:,k)=imag(fun(x1))/h;
    end

end
```