AOLoopAR function

```
function [var_eps] = AOloopAR(G,H, covariance_phi, sigma_e, A, Cw, K,
 phi_sim)
% Variance calculation of an AO system in the closed-loop
 configuration
% with a Kalman filter involved
% IN
% G      : measurement matrix
% H      : influence matrix mapping the wavefront on the mirror
% covariance_phi  : covariance matrix of the turbulence wavefront
% sigma_e : measurement noise parameter for determining its covariance
% A : A matrix from the state-space model
% Cw : Covariance matrix of the process noise
% K : Stationary Kalman filter gain
% phi_sim : simulation data for the wavefront
% OUT
% var_eps : variance of the residual wavefront after taking N_t points
% within the closed-loop operation

% dimension lifted wavefront
n_H = size(H,1);

% dimension lifted sensor slopes
n_G = size(G,1);

% Number of sample points for phi_sim
T = length(phi_sim);

u = zeros(n_H,T);

% epsilon matrix
eps_k = zeros(size(phi_sim,1),T);

eps_kplus1k = zeros(size(phi_sim,1),T);

% epsilon matrix with mean removed:
eps_mean_removed_k = zeros(size(phi_sim,1),T);

% Constructing a vector of all the variance values for eps
var_eps = zeros(T,1);

% An assumption is that H is full rank.

% s(k) = G*eps(k) + e(k)
% Hence s(k) = G*phi(k) - G*H*u(k-1) + e(k)

% u(k) = 0 for k < 1

% eps(k+1|k) = Ks(k) + A*H*u(k-1) - H*u(k) + (A - KG)eps(k|k-1)
% eps(k+1|k) = K*G*phi(k) - K*G*H*u(k-1) + K*e(k) + A*H*u(k-1) -
 H*u(k) + (A - KG)eps(k|k-1)

% First slope value, no input applied before
s_k = G*phi_sim(:,1) + sigma_e*randn(n_G,1);

% Initial input
u(:,1) = H\(K*s_k);

% eps(1|0) = 0;
%eps_kplus1k = eps(k+1|k)
eps_kplus1k(:,1) = K*G*phi_sim(:,1) + K*sigma_e*randn(n_G,1) -
 H*u(:,1);

% Initial eps(k)
eps_k(:,1) = phi_sim(:,1);
eps_mean_removed_k(:,1) = eps_k(:,1) - mean(eps_k(:,1));

var_eps(1) = var(eps_mean_removed_k(:,1));

for k = 2:T
    % Slope value
    s_k = G*phi_sim(:,k) - G*H*u(:,k-1) + sigma_e*randn(n_G,1);

    % Optimum input value
    u(:,k) = H\(K*s_k + (A - K*G)*eps_kplus1k(:,k-1) + A*H*u(:,k-1));

    % Next eps(k+1|k) value
    eps_kplus1k(:,k) = K*s_k + A*H*u(:,k-1) - H*u(:,k) + (A -
 K*G)*eps_kplus1k(:,k-1);

    % eps(k) value = phi(k) - Hu(k-1)
    eps_k(:,k) = phi_sim(:,k) - H*u(:,k-1);
    eps_mean_removed_k(:,k) = eps_k(:,k) - mean(eps_k(:,k));
    var_eps(k) = var(eps_mean_removed_k(:,k));
end

var_eps = mean(var_eps);

end
```

Published with MATLAB® R2018a