LMS Algorithm

- SD requires knowledge of R and P. In many applications these second order statistics are unknown.
- Least mean square (LMS) algorithm
  - Make estimate of gradient (noisy gradient descent algorithm)
  - Estimate based on one observation (u(k), d(k))

$$\hat{\nabla} J(w(k)) = -e(k)u(k)$$

$$w(k+1) = w(k) + \mu e(k)u(k)$$

LMS Algorithm Properties

- Steepest Descent and LMS algorithm convergence depends on step size μ and eigenvalues of R.
- LMS algorithm is simple to implement.
- LMS algorithm convergence is relatively slow.
- Tradeoff between convergence speed and excess MSE.
- LMS algorithm can track training data that is time varying.

LMS Convergence Behavior

- **Assumptions: u(n) iid sequence, u(n) independent of d(n-k), k > 0, d(n) independent of y(n-k), k>0, u(n) and d(n) are jointly Gaussian.**
- **Mean convergence analysis: Let $e^*(k) = d(k) - w^{*T} u(k)$, denote error from optimal weight at time k.**
  - **$E(c(k+1)) = (I - \mu R) E(c(k)) + \mu E(u(k)e^*(k))$**
  - **Asymptotically assuming step size is chosen correctly, then $\lim_k E(c(k)) = 0$ and $E(w(k))$ converges to w^***
- **Mean squared analysis studies cost function J(w(k)). Note $tr(R) > \lambda_{max}$ and more conservative bound given by $0 < \mu < 2 / tr(R)$.**

Mean Squared Error Analysis

- $J(w)$ consist of transient and steady state effects. Let $J_s(w)$ be steady state effects

$$J_s(w) = J_{min} + J_{excess}$$

- Transient effects depend on eigenvalues. Define λ_{avg} as the average of eigenvalues = tr(R)/n. Rate of decay is exponential with $r_{avg} = (1-\mu\,\lambda_{avg})^2 = \exp(-1/\tau)$. It takes roughly 4 time constants, $4\tau \approx 2n/(\mu\,\mathrm{tr}(R))$.
- $J_{excess} \approx \mu\, J_{min}\, \mathrm{tr}(R)/2$

Step size tradeoff

- Larger step size μ quicker convergence, but more excess mean squared error.
- Smaller step size μ slower convergence, but less excess mean squared error.
- Misadjustment (dimensional quantity) proportional to step size

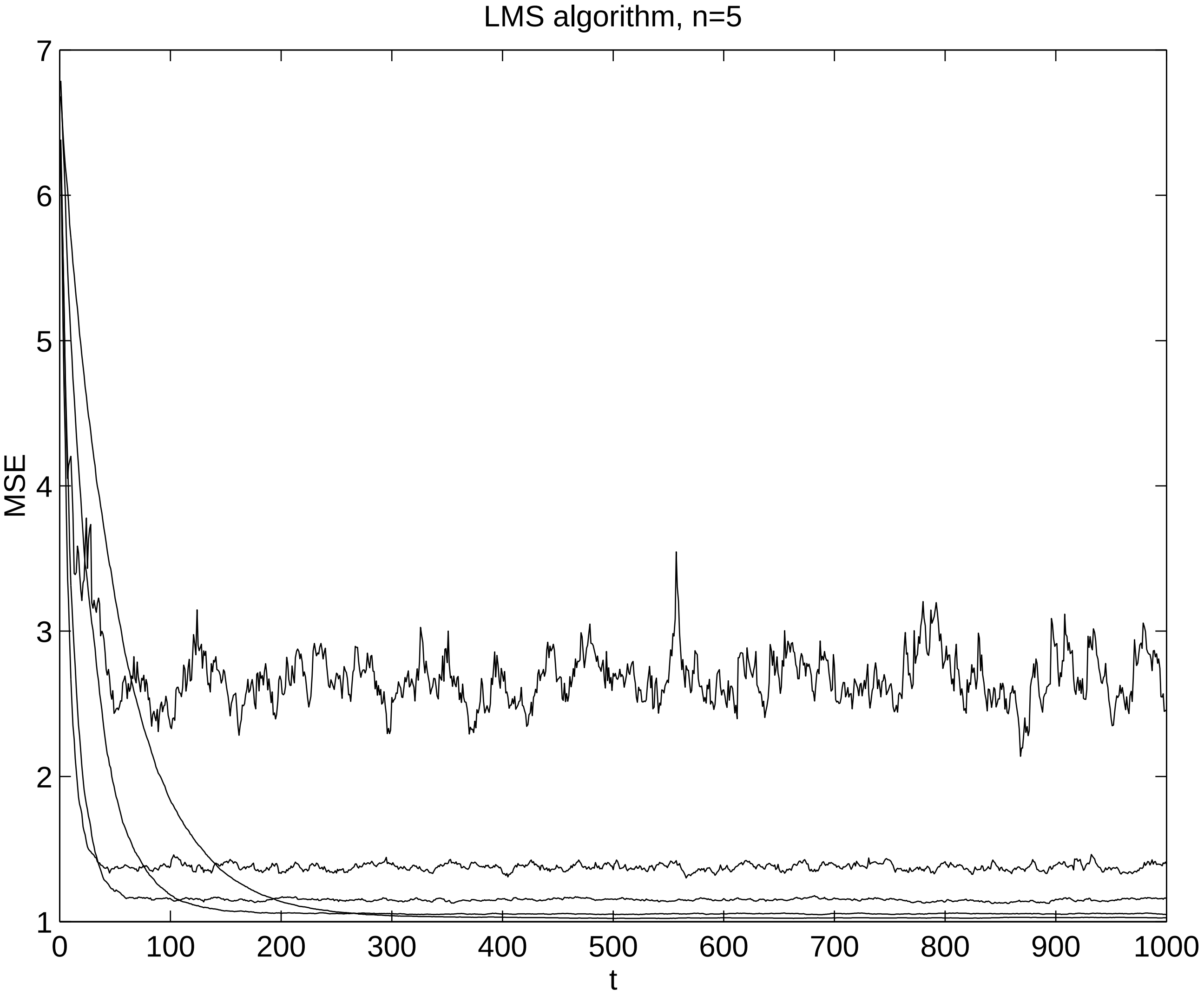
$$\mathcal{M} = J_{excess} / J_{min} = .5 \mu \, \mathrm{tr}\,(R)$$

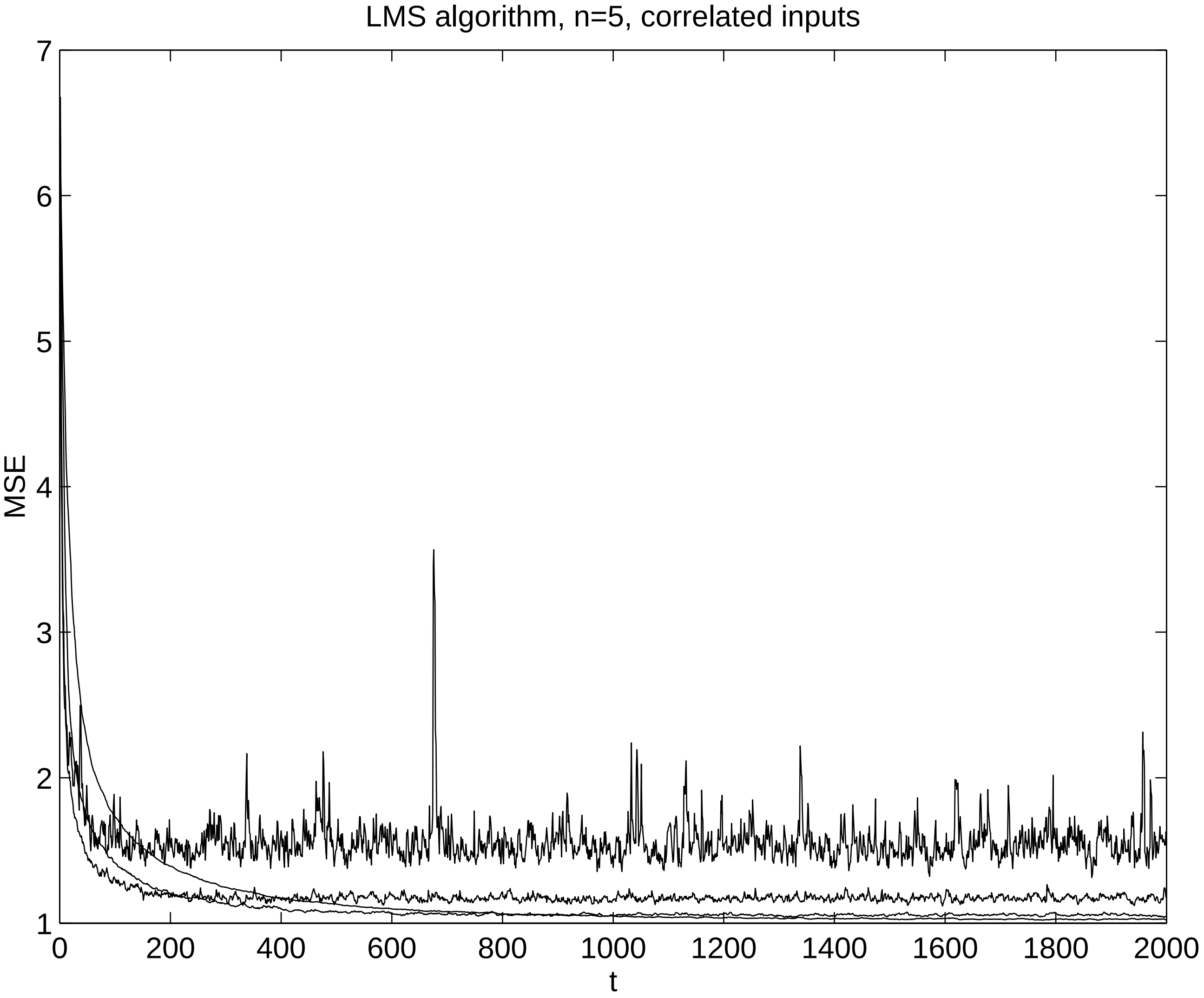
Simulation Study

- Inputs drawn from zero mean Gaussian processes that are widesense stationary
  - Input components are uncorrelated
  - Input components satisfy $E(x_i (k) x_j(k)) = .8^{|i-j|}$
- Step size for uncorrelated case μ= .2, .1, .05, .02, and .01. Same step size for correlated case except μ= .2 which did not converge
- Additive noise is zero mean unit variance Gaussian process independent of x(k)
- Simulations run 100 times and averaged

Simulation Observations

- **Analysis accurate for uncorrelated case (equal eigenvalues).**
- **Step size bound for LMS, $0 < \mu < 2 / \mathrm{tr}(R)$ for uncorrelated case.**
- **For correlated case, eigenvalues are 3.64, .79, .29, .16, and .12. Condition number for matrix is 30. Analysis not accurate for this case.**


LMS algorithm, n=5
MSE
t
1
2
3
4
5
6
7
0
100
200
300
400
500
600
700
800
900
1000


LMS algorithm, n=5, correlated inputs
MSE
t
1
2
3
4
5
6
7
0
200
400
600
800
1000
1200
1400
1600
1800
2000

Other Iterative Algorithms

- LMS algorithm with variable step size:

$$w(k+1) = w(k) + \mu(k)e(k)x(k)$$

When step size $\mu(k) = \mu/k$ algorithm converges almost surely to optimal weights.

- Conjugate gradient (CG) algorithm: Gradients and line search used to form CGs. Algorithm converges in n steps.
- Newton's algorithm: Let g(n) be gradient and H(n) be Hessian of w(n) then approximate energy function by

$$J(w) \approx J(w(n)) + (w-w(n))^T g(n) + .5\,(w-w(n))^T H(n)(w-w(n))$$

Take gradient of approximation and set to zero to get

$$w(n+1) = w(n) + H(n)^{-1}\, g(n)$$

Algorithm involves inverting Hessian matrix (costly).

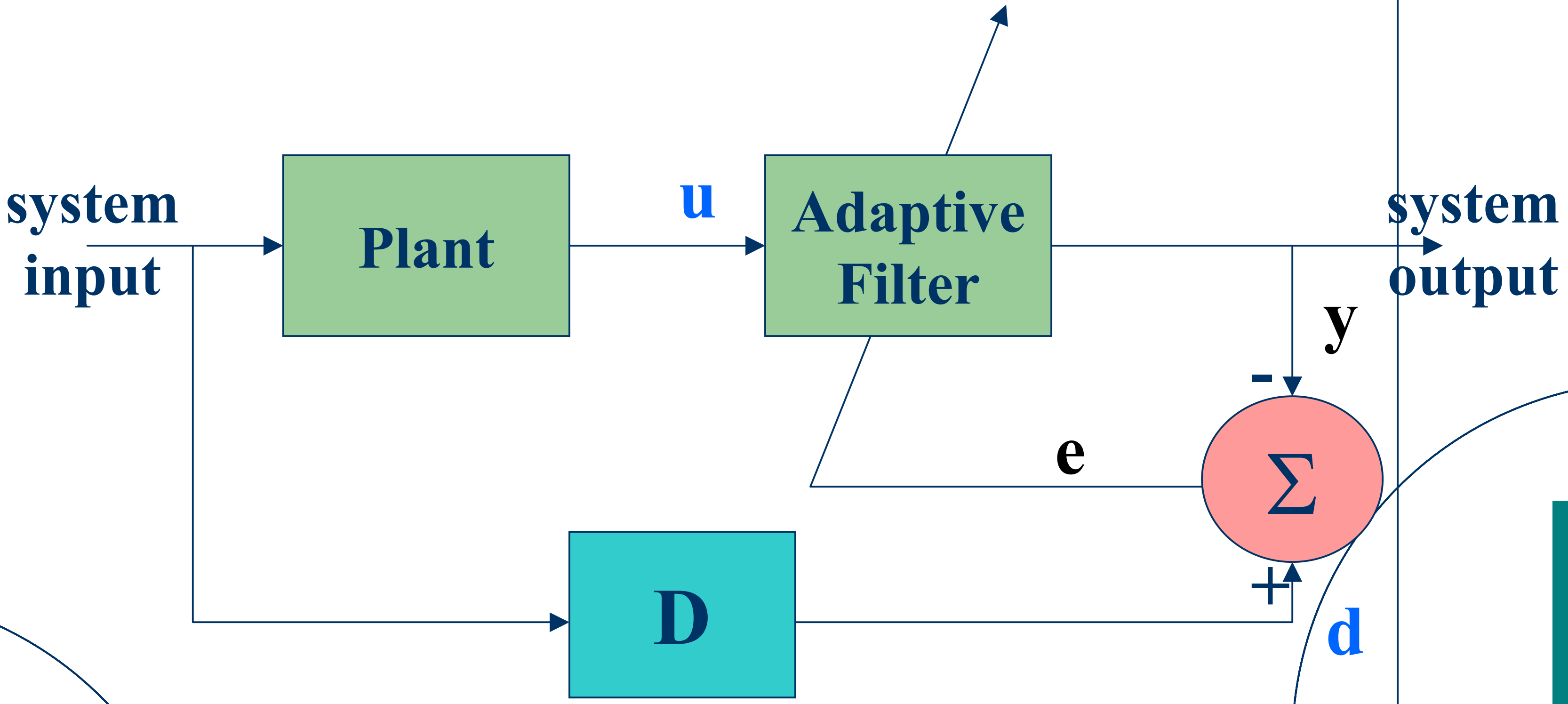
Iterative Algorithm Comments

- Algorithms based on descending energy surface by examining first and second derivatives.
- Tradeoffs between algorithm complexity and convergence speed.
- Can use other cost functions besides quadratic cost functions: Absolute error, Minkowski error, entropy function.

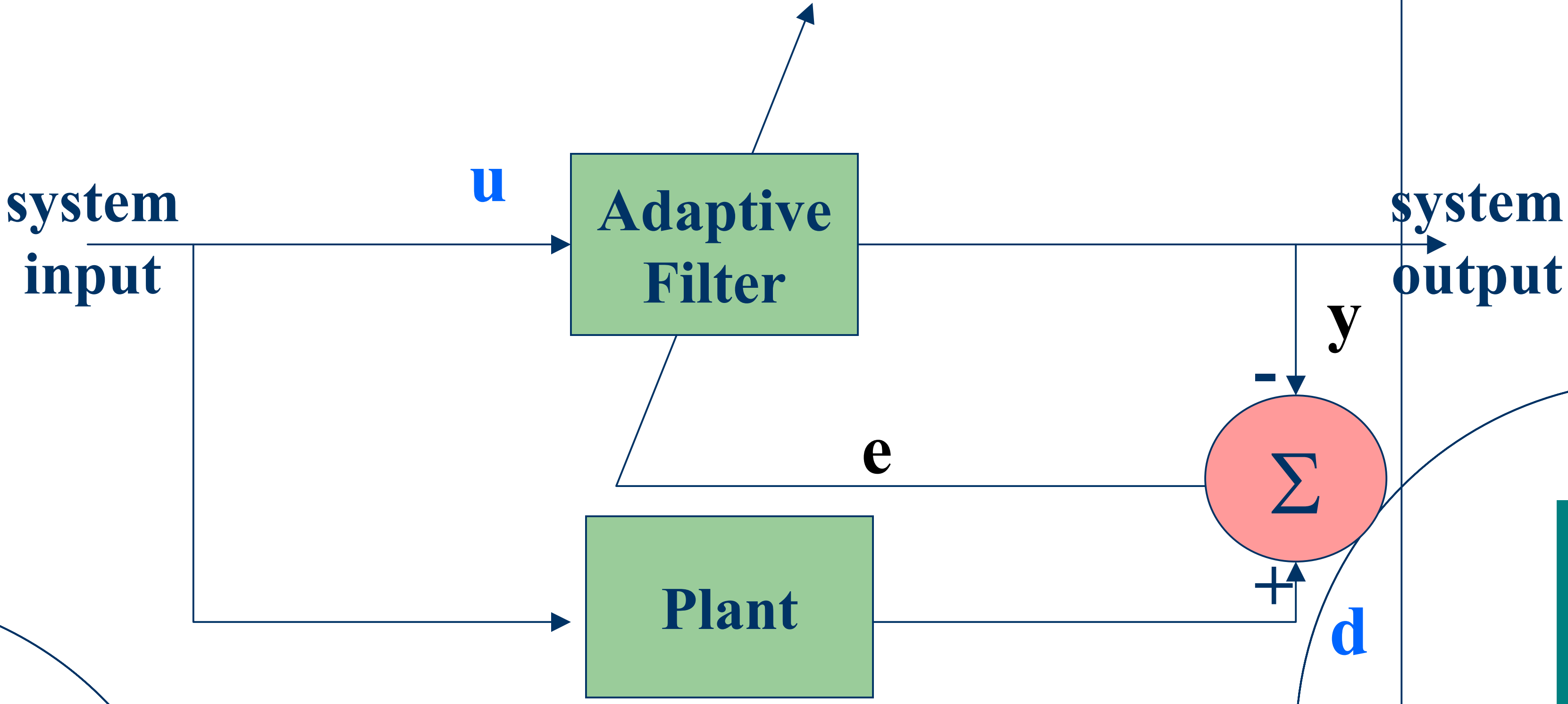
Linear Filter Applications

- Inverse Modeling: Channel Equalization
- System Identification: Plant modeling
- Prediction
- Adaptive Interference Cancellation: Echo Cancellation
- Adaptive Beamforming
  - Radar
  - Sonar
  - Speech enhancement

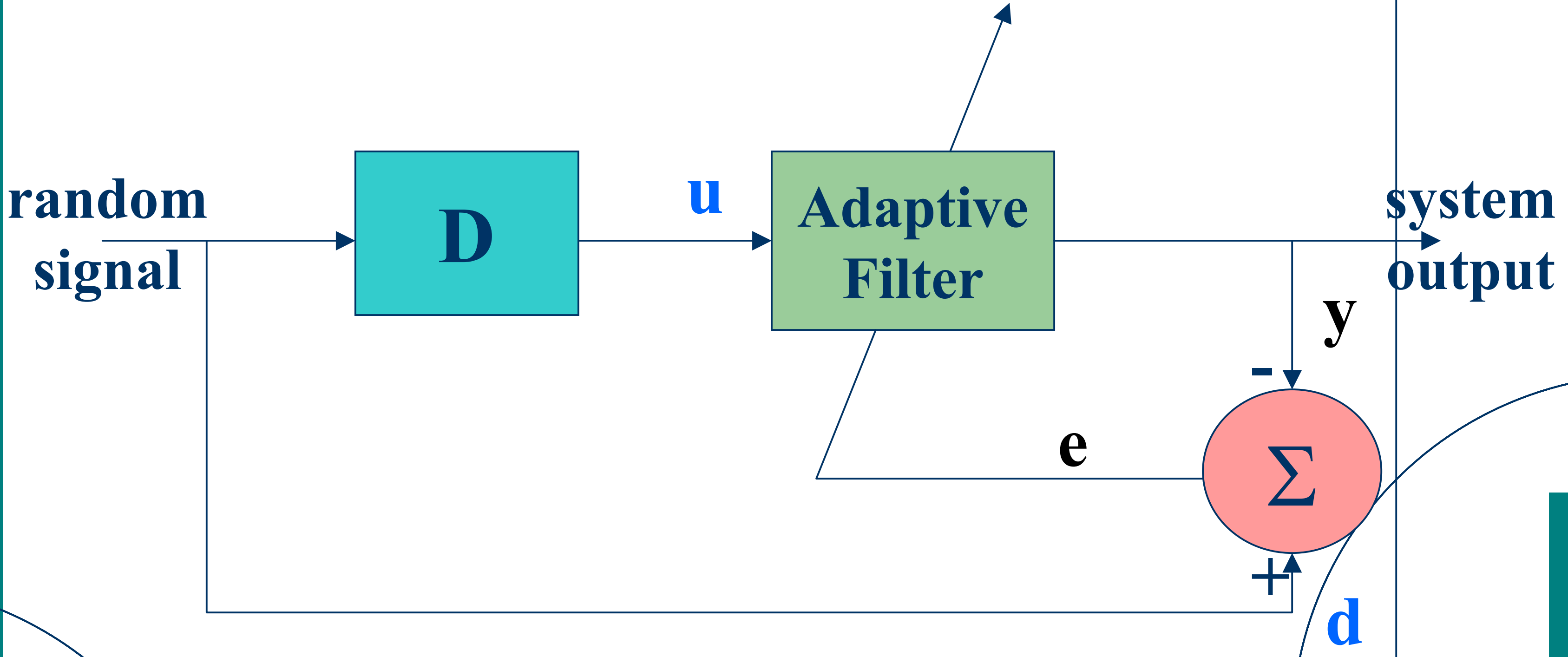
Inverse Modeling



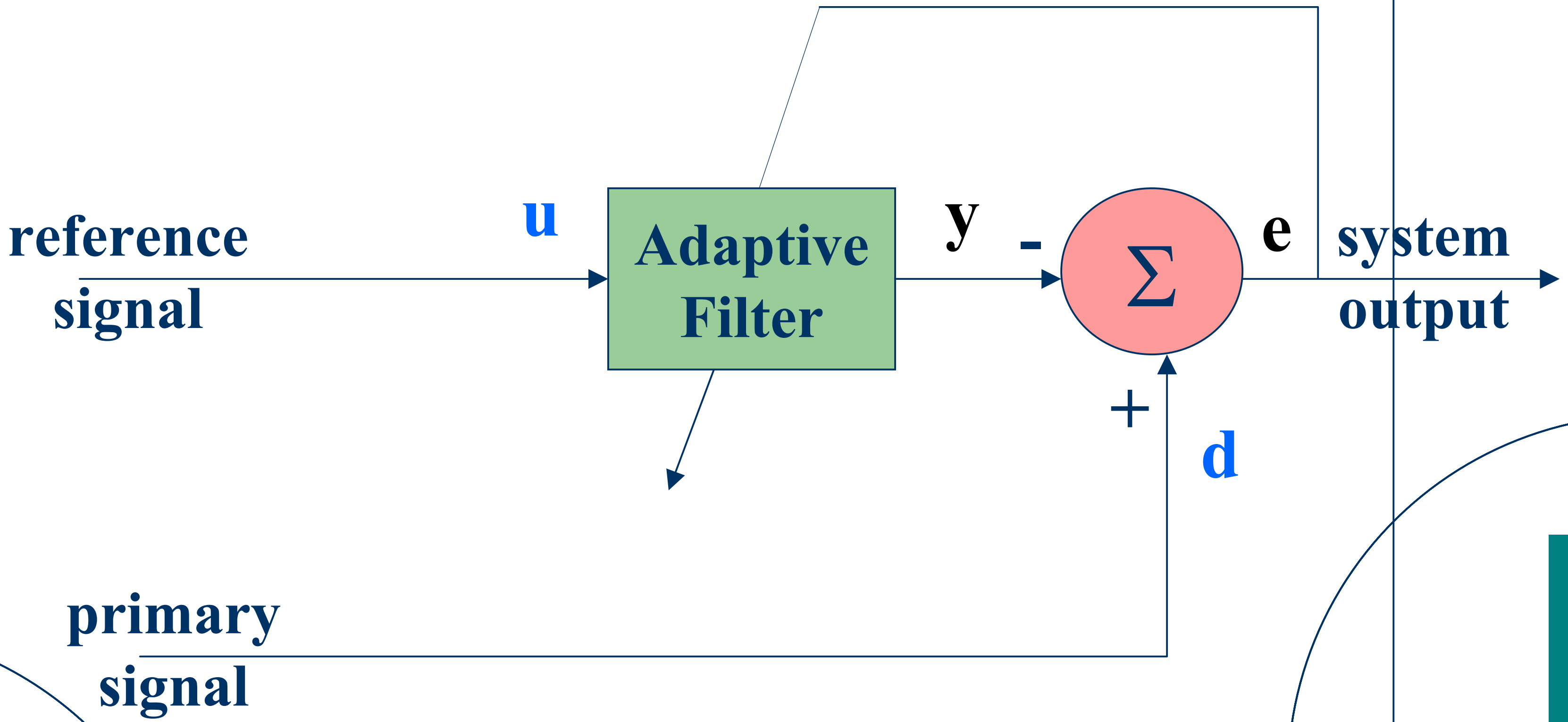
System Identification



Prediction



Interference Cancellation



reference signal

u

Adaptive Filter

y

-

Σ

e

system output

+

d

primary signal