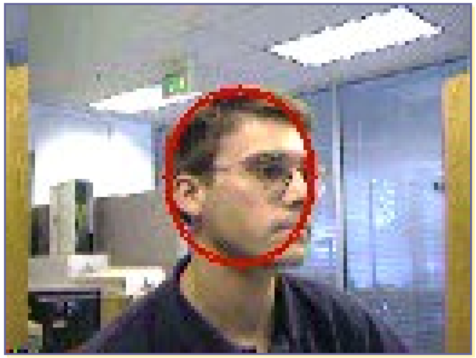
Part 2: Kalman Filtering

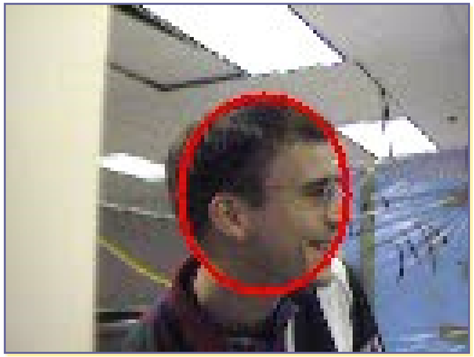
COS 323

On-Line Estimation

- Have looked at "off-line" model estimation: all data is available
- For many applications, want best estimate immediately when each new datapoint arrives
  - Take advantage of noise reduction
  - Predict (extrapolate) based on model
- Additionally: Take advantage of multiple sensors (in a principled way)
- Applications: controllers, tracking, …

Face Tracking





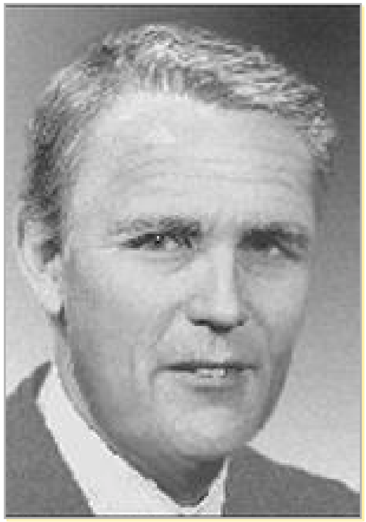
Birchfield

On-Line Estimation

- Have looked at “off-line” model estimation: all data is available
- For many applications, want best estimate immediately when each new datapoint arrives
  - Take advantage of noise reduction
  - Predict (extrapolate) based on model
  - Applications: controllers, tracking, …
- How to do this without storing all data points?

Kalman Filtering

- Assume that results of experiment are noisy measurements of "system state"
- Use a model of how system evolves
- Combine system model and observations to deduce "true" state
- Prediction / correction framework



Rudolf Emil Kalman

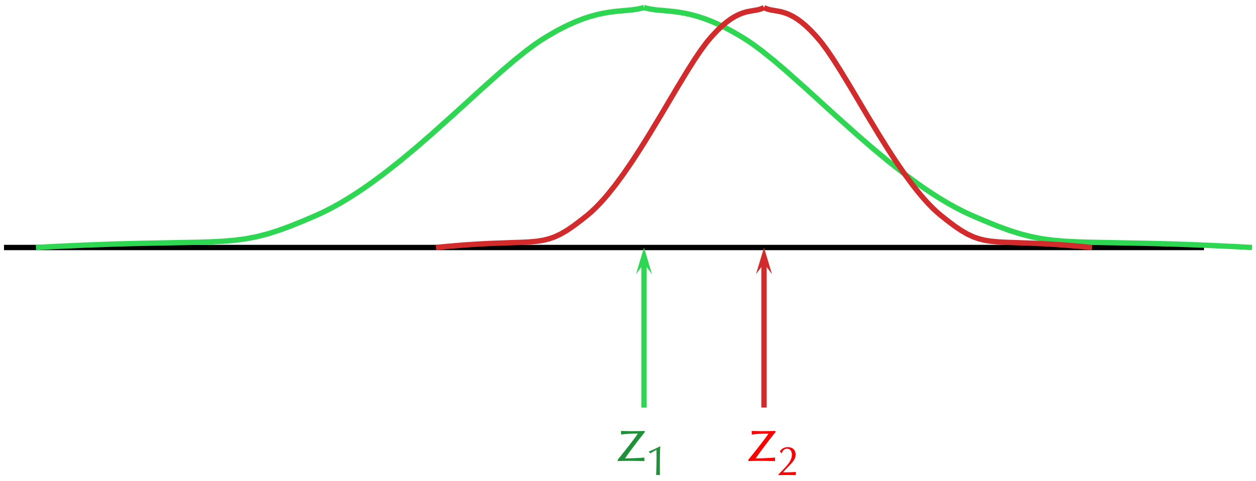
Acknowledgment: much of the following material is based on the SIGGRAPH 2001 course by Greg Welch and Gary Bishop (UNC)

Simple Example

- Measurement of a single point z_1
- Variance σ_1^2 (uncertainty σ_1)
- Best estimate of true position $\hat{x}_1 = z_1$
- Uncertainty in best estimate $\hat{\sigma}_1^2 = \sigma_1^2$

Simple Example

- Second measurement z_2, variance $\sigma_2{}^2$
- Best estimate of true position?



Simple Example

- Second measurement z_2, variance σ_2^2
- Best estimate of true position: weighted average

$$\hat{x}_2 = \frac{\frac{1}{\sigma_1^2} z_1 + \frac{1}{\sigma_2^2} z_2}{\frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2}}$$

$$= \hat{x}_1 + \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}\left(z_2 - \hat{x}_1\right)$$

- Uncertainty in best estimate: $\hat{\sigma}_2^2 = \dfrac{1}{\frac{1}{\hat{\sigma}_1^2} + \frac{1}{\sigma_2^2}}$

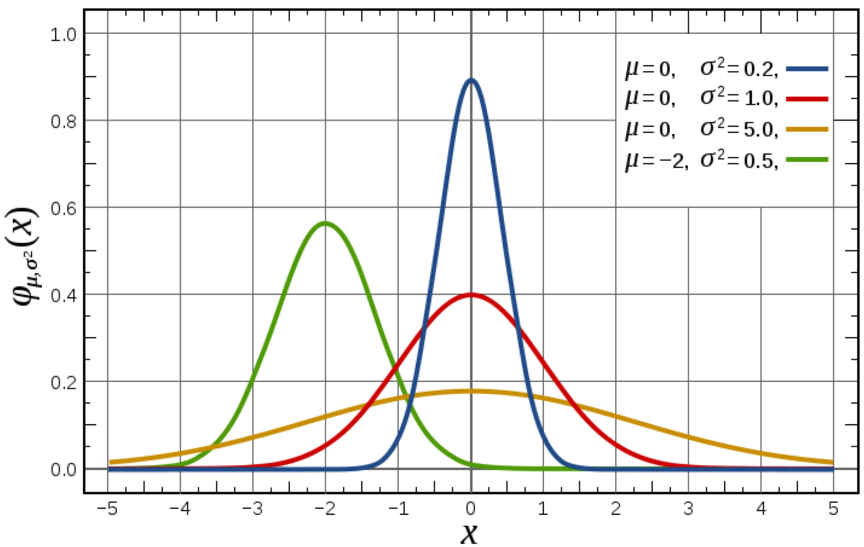
Online Weighted Average

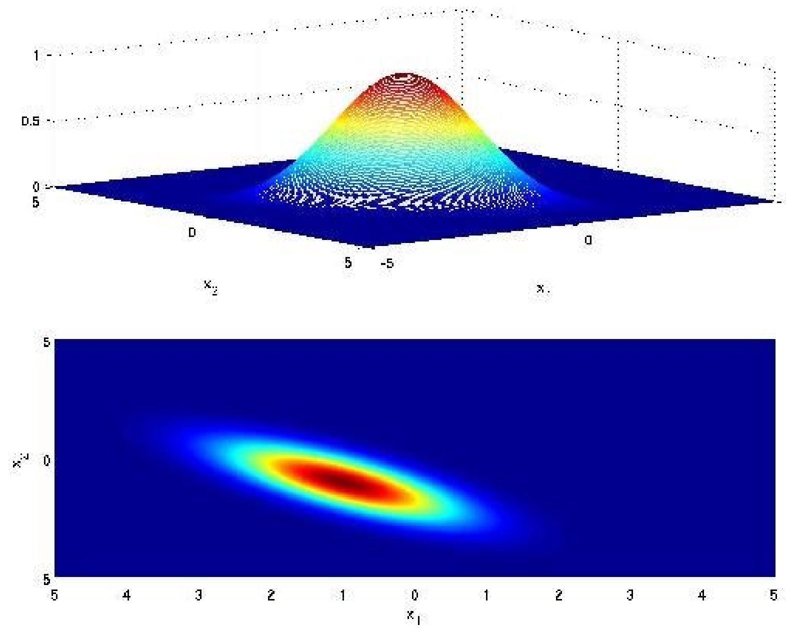
- Combine successive measurements into constantly-improving estimate
- Uncertainty usually decreases over time
- **Only need to keep current measurement, last estimate of state, and uncertainty**

Terminology

- In this example, position is *state* (in general, any vector)
- State can be assumed to evolve over time according to a *system model* or *process model* (in previous example, “nothing changes”)
- Measurements (possibly incomplete, possibly noisy) according to a *measurement model*
- Best estimate of state $\hat{x}$ with covariance P

Gaussian Review





Linear Models

- For "standard" Kalman filtering, everything must be linear
- System model:

$$x_k = \Phi_{k-1} x_{k-1} + \xi_{k-1}$$

- The matrix Φ_k is *state transition matrix*
- The vector ξ_k represents *additive noise*, assumed to have mean $\mathbf{0}$ and covariance Q

$$\mathbf{x}_k = \begin{bmatrix} x \\ dx/dt \end{bmatrix}, \quad \Phi_k = \begin{bmatrix} 1 & \Delta t_k \\ 0 & 1 \end{bmatrix}$$

Linear Models

- Measurement model:

$$z_k = H_k x_k + \mu_k$$

- Matrix H is *measurement matrix*
- The vector μ is *measurement noise*, assumed to have mean **0** and covariance R

Position + Velocity Model

$$x_k = \Phi_{k-1} x_{k-1} + \xi_{k-1}$$

$$\mathbf{x}_k = \begin{bmatrix} x \\ {dx}/{dt} \end{bmatrix}$$

$$z_k = H_k x_k + \mu_k$$

$$\Phi_k = \begin{bmatrix} 1 & \Delta t_k \\ 0 & 1 \end{bmatrix}$$

$$H = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

Prediction/Correction

- Multiple values around at each iteration:
  - x'_k is prediction of new state on the basis of past data (i.e., our "a priori" estimate)
  - z'_k is predicted observation
  - z_k is new observation
  - $\hat{x}_k$ is new estimate of state ("a posteriori")

Prediction/Correction

- 1: Predict new state

$$x'_k = \Phi_{k-1}\hat{x}_{k-1}$$

$$P'_k = \Phi_{k-1}P_{k-1}\Phi^{\mathrm{T}}_{k-1} + Q_{k-1}$$

$$z'_k = H_k x'_k$$

- 2: Correct to take new measurements into account

$$\hat{x}_k = x'_k + K_k\left(z_k - H_k x'_k\right)$$

$$P_k = \left(I - K_k H_k\right)P'_k$$

Kalman Gain

$$\hat{x}_k = x'_k + K_k\left(z_k - H_k x'_k\right)$$
$$P_k = \left(I - K_k H_k\right)P'_k$$

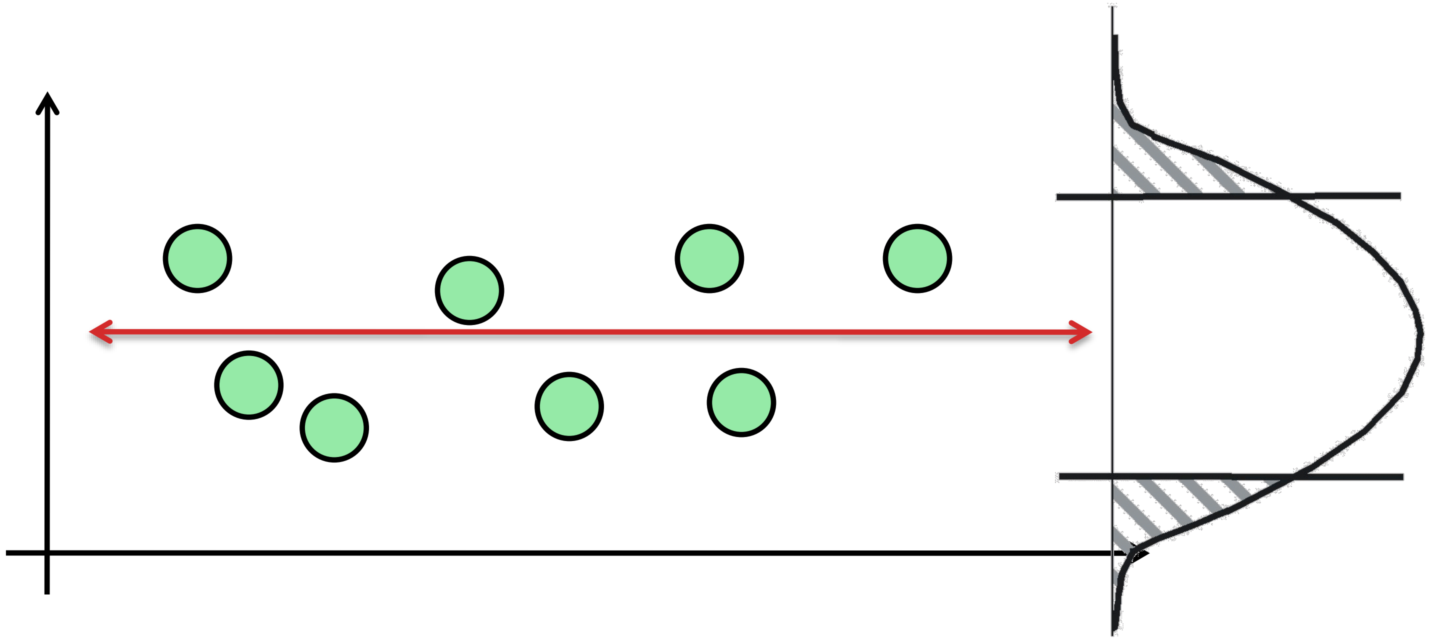
- K is weighting of process model vs. measurements, chosen to minimize P_k:

$$K_k = P'_k H_k^{\mathrm{T}}\left(H_k P'_k H_k^{\mathrm{T}} + R_k\right)^{-1}$$

- Compare to what we saw earlier: $\dfrac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}$

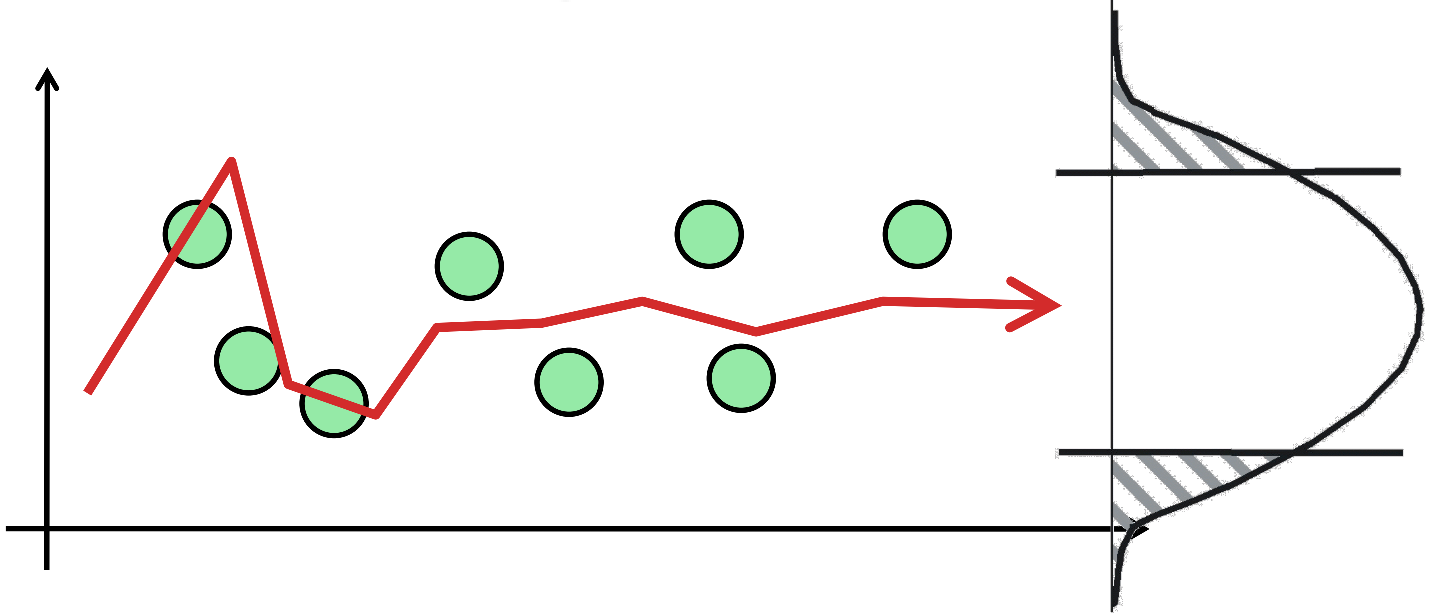
Example: Estimate Random Constant

offline case: compute μ, σ^2



Example: Estimate Random Constant

online case: compute x_k (μ_k) P_k (σ_k^2)



Example: Estimate Random Constant

Predict:

$$x'_k = \Phi_{k-1}\hat{x}_{k-1} \text{ becomes } x'_k = \hat{x}_{k-1}$$

$$P'_k = \Phi_{k-1}P_{k-1}\Phi^{\mathrm{T}}_{k-1} + Q_{k-1} \text{ becomes } P'_k = P_{k-1} + Q_{k-1}$$

$$z'_k = H_k x'_k + \mu_k \text{ becomes } z'_k = x'_k + \mu_k$$

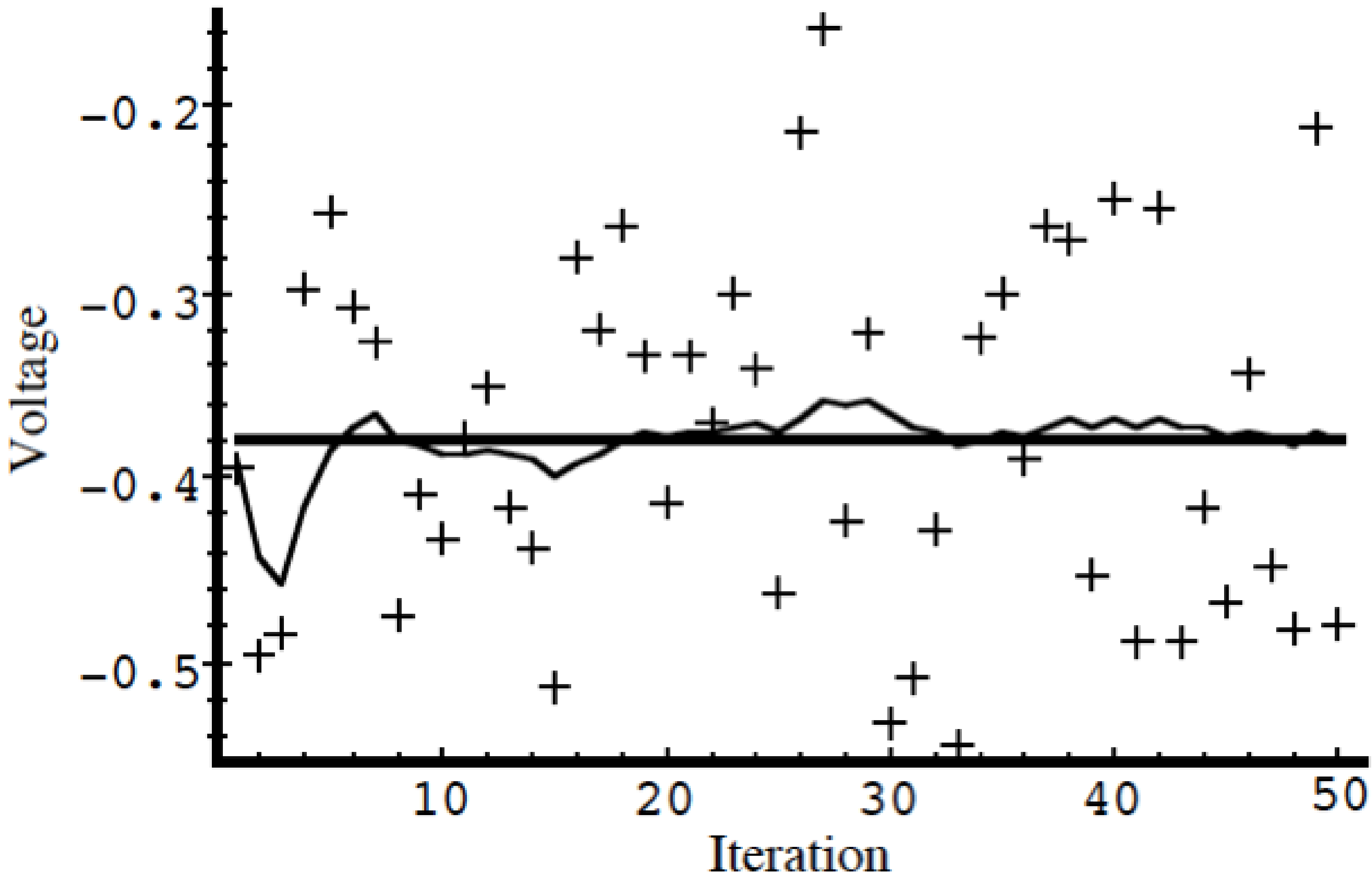
Update:

$$K = P'_k / (P'_k + R)$$

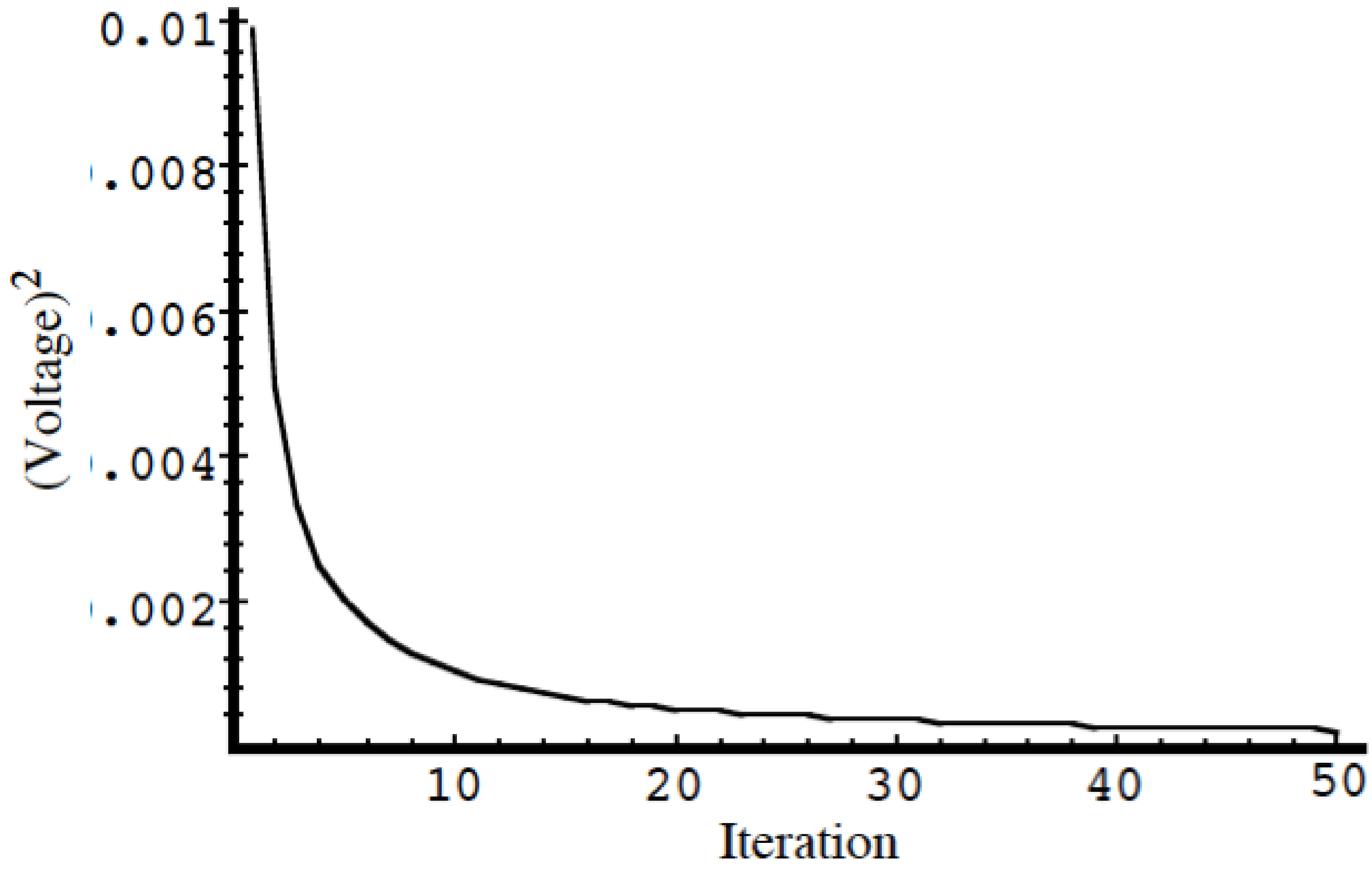
$$\hat{x}_k = x'_k + K_k\left(z_k - x'_k\right)$$

$$P_k = \left(I - K_k\right)P'_k$$

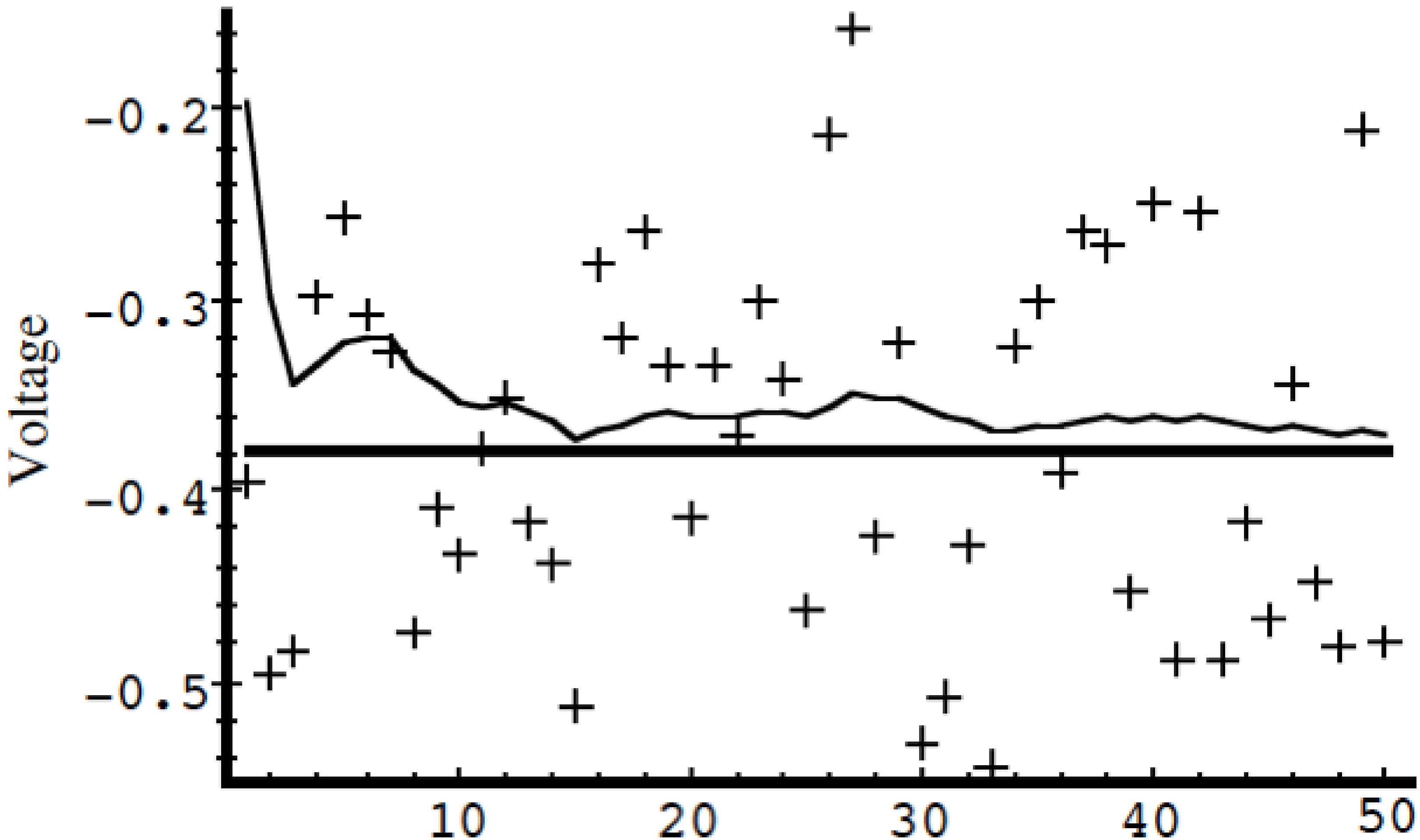
Simulation: R selected to be true measurement error variance



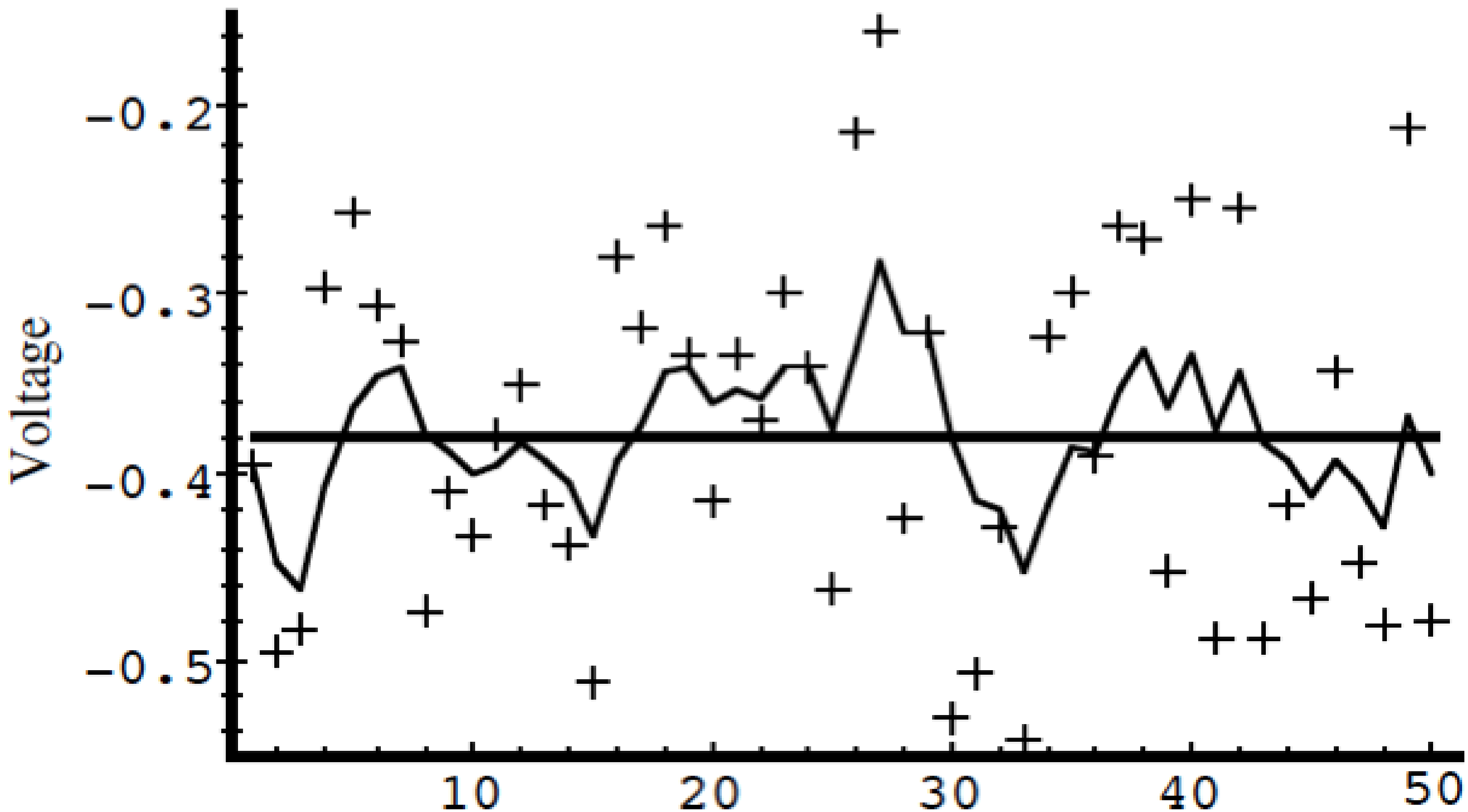
Pk decreasing with each iteration



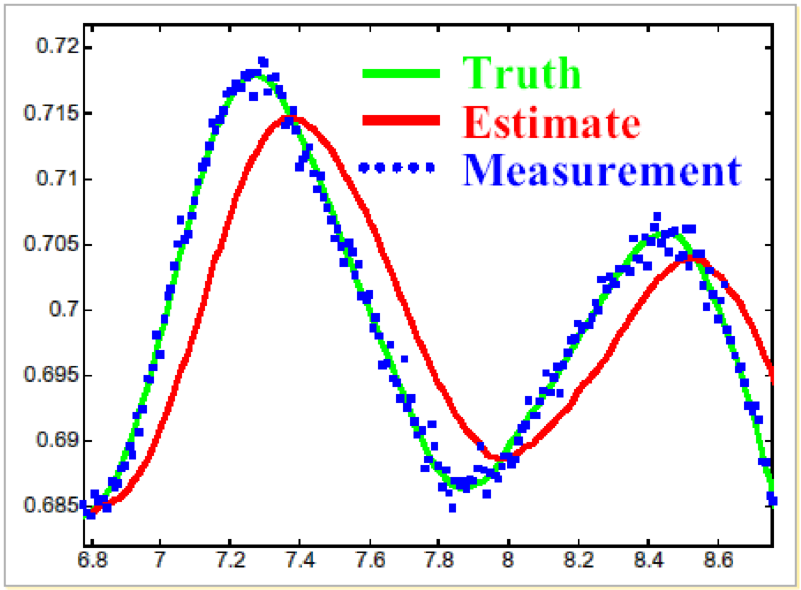
Simulation:
R overestimates measurement error



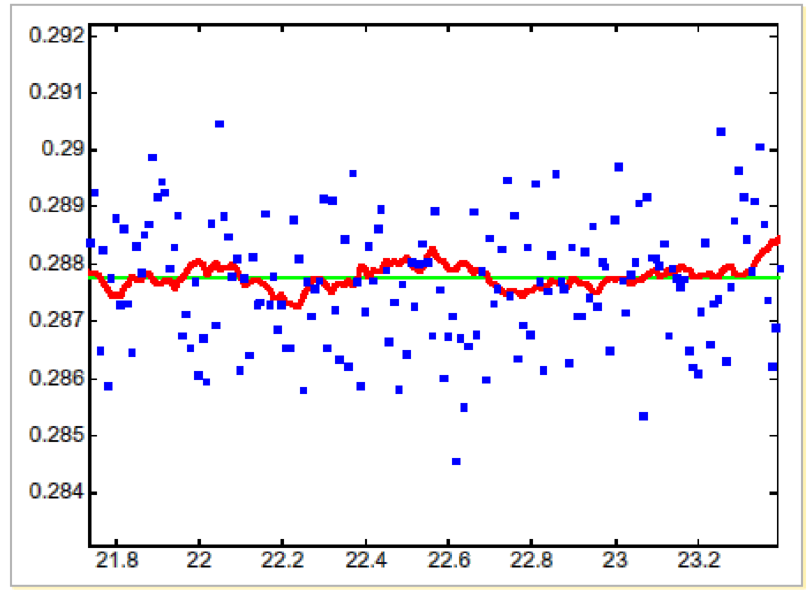
Simulation:
R underestimates measurement error



Results: Position-Only Model



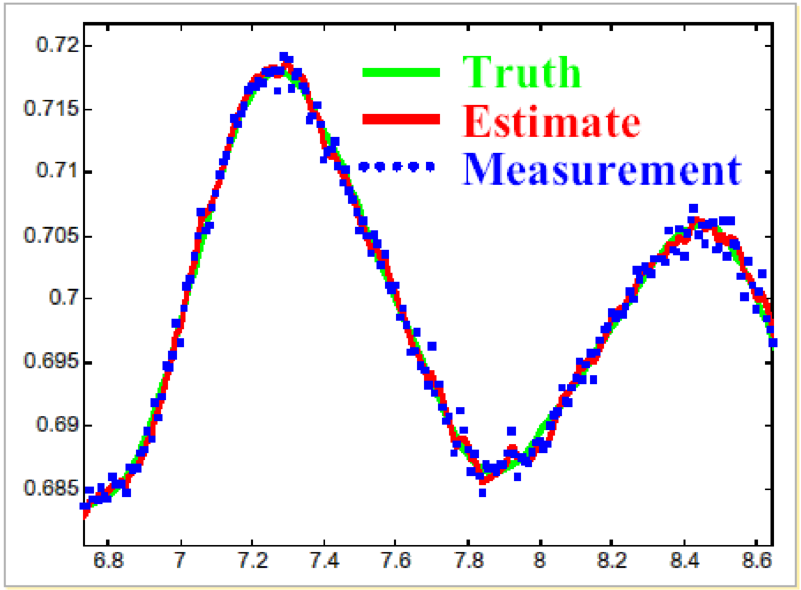
Moving



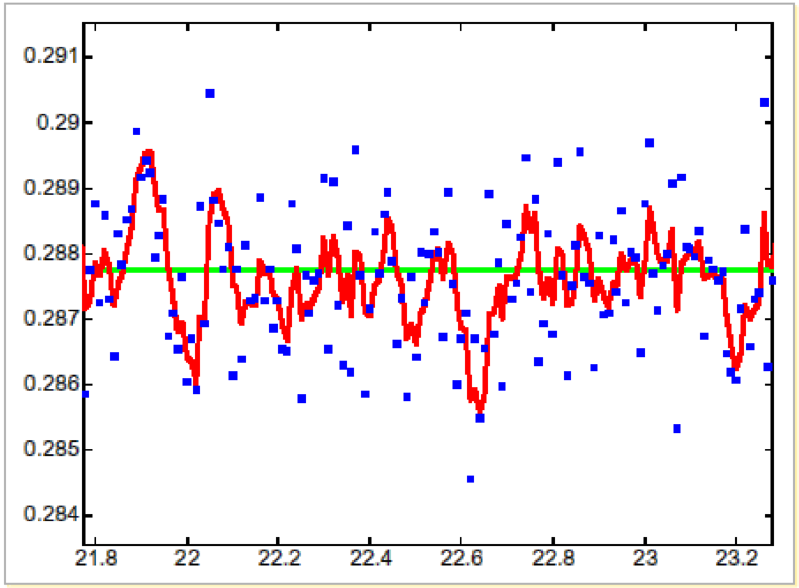
Still

[Welch & Bishop]

Results: Position-Velocity Model



Moving



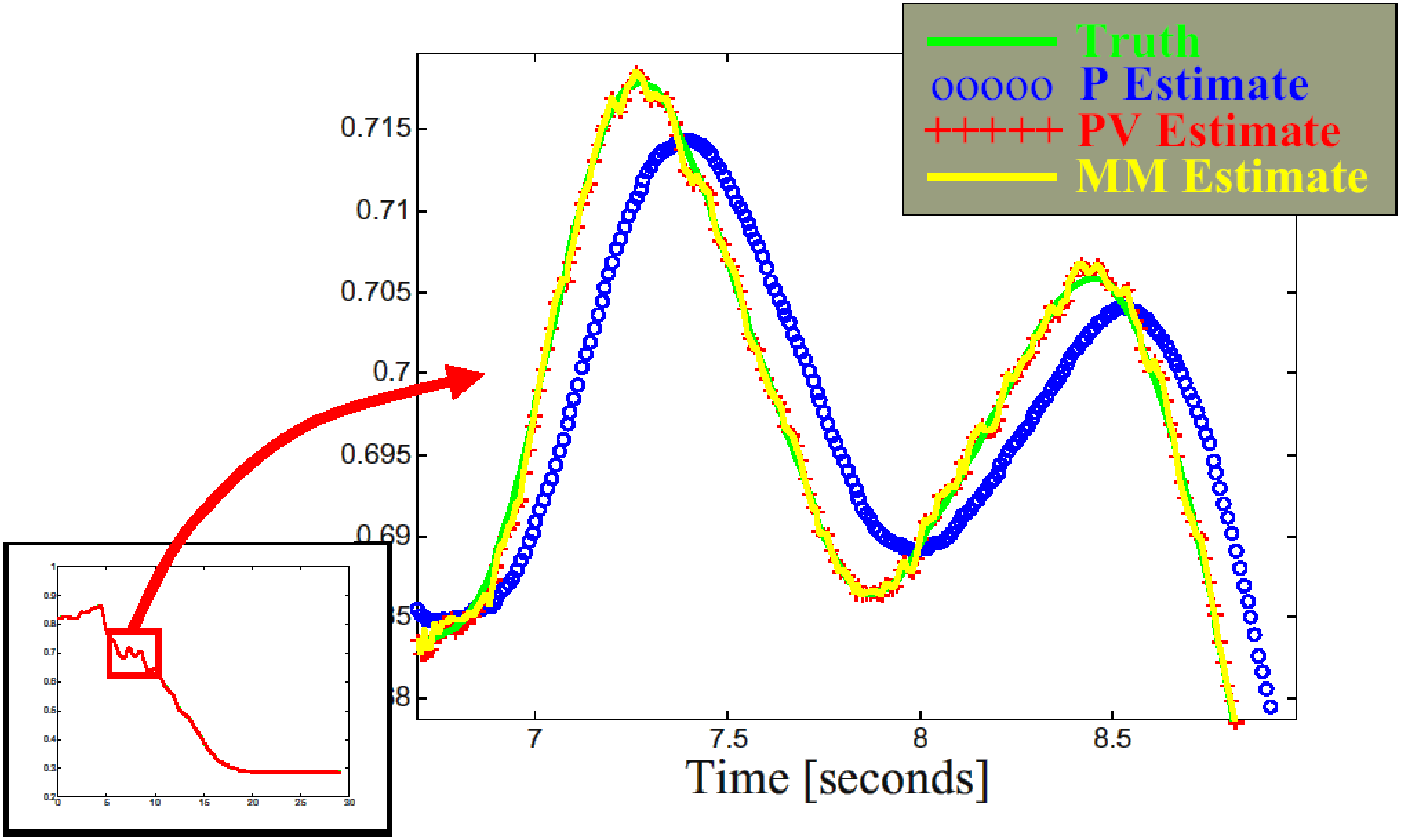
Still

[Welch & Bishop]

Extension: Multiple Models

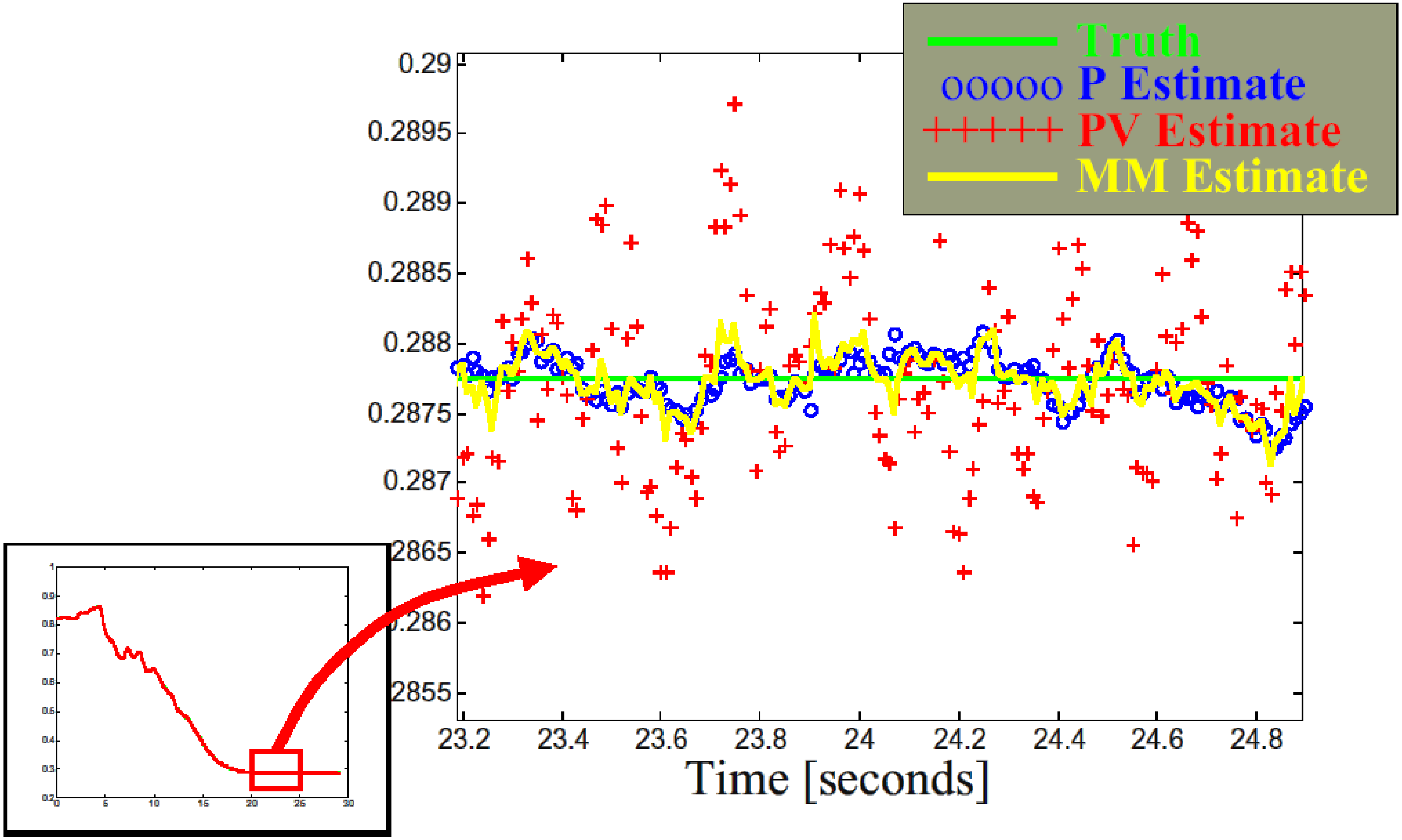
- Simultaneously run many KFs with different system models
- Estimate probability each KF is correct
- Final estimate: weighted average

Results: Multiple Models



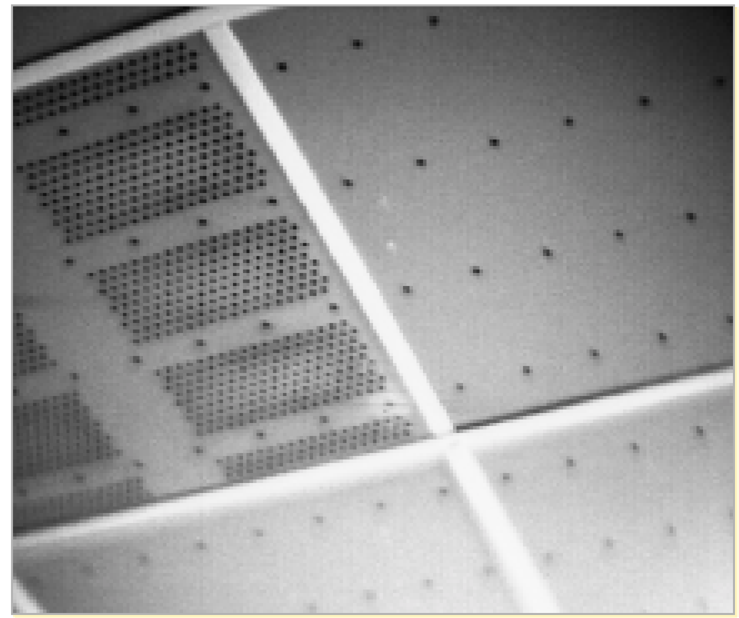
[Welch & Bishop]

Results: Multiple Models



[Welch & Bishop]

UNC HiBall



- 6 cameras, looking at LEDs on ceiling
- LEDs flash over time

[Welch & Bishop]

Extension: Nonlinearity (EKF)

- HiBall state model has nonlinear degrees of freedom (rotations)
- Extended Kalman Filter allows nonlinearities by:
  - Using general functions instead of matrices
  - Linearizing functions to project forward
  - Like 1st order Taylor series expansion
  - Only have to evaluate Jacobians (partial derivatives), not invert process/measurement functions

Other Extensions & Related Concepts

- Using known system input (e.g. actuators)
- Using information from both past and future
- Non-Gaussian noise and particle filtering
- Hidden Markov Models: discrete state space
- Read the Welch & Bishop tutorial on course webpage