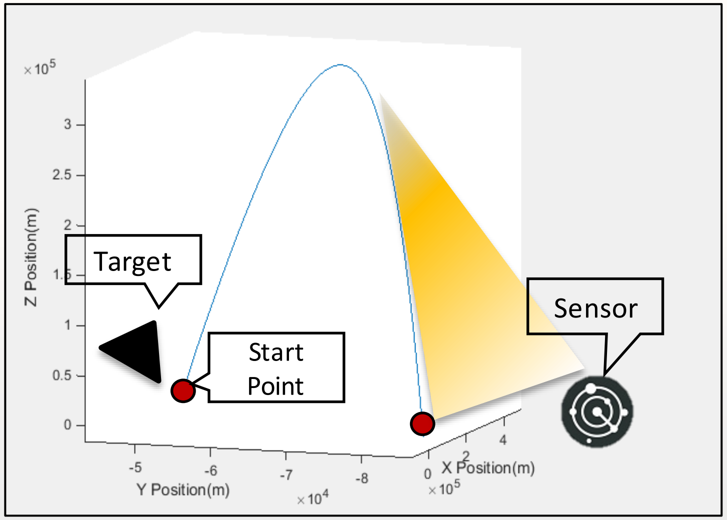
Data-Driven Ballistic Coefficient Learning for Future State Prediction of High-Speed Vehicles

**Kyungwoo Song, Sang-Hyeon Kim, Jinhyung Tak,
Han-Lim Choi, Il-Chul Moon**

CONTENTS

INTRODUCTION–ESTIMATION

Observation with Sensor



Observation value

t

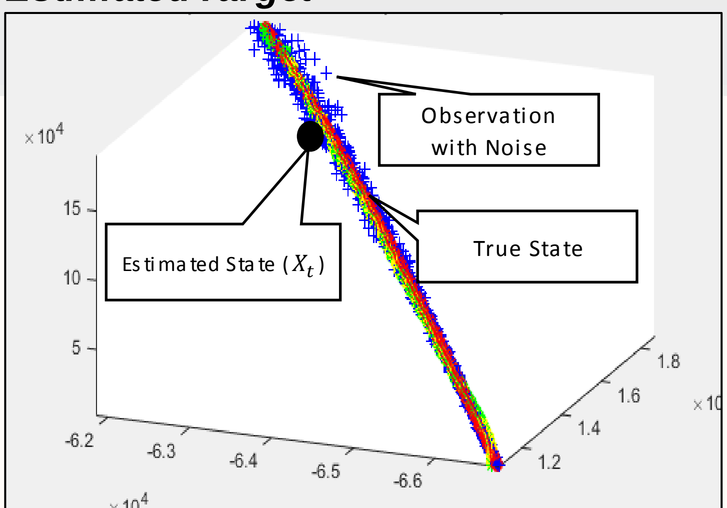
$Rang\quad ,Azim_u\quad angl\quad ,Elevat\quad angl\quad e$

$Rang\ 2^e, Azim_u\quad angl\ 2, Elevat\quad angl\ 2$

$Rang\ 3^e, Azim\quad ang\ 3, Eleva\quad ang\ 3$

… …

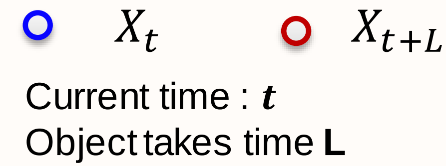
Estimated Target



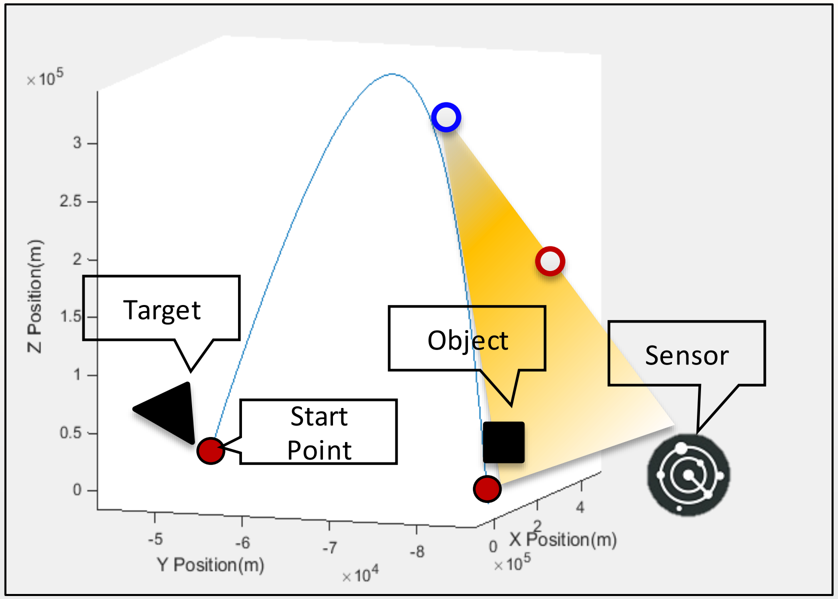
High-Speed Vehicle Target Tracking

- Observation , Dynamic Equations
- Position, Velocity, Acceleration of Target
- EKF, UKF, Particle Filter
- Traditional Problem

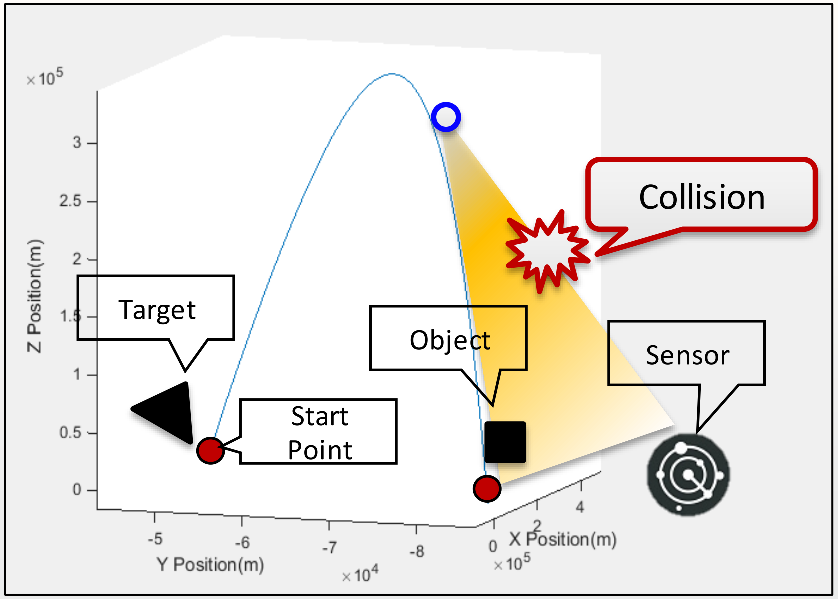
INTRODUCTION–PREDICTION



Shoot at X_t



Shoot at X_{t+L}



High-Speed Vehicle Target Prediction

- Predict future State $(t + L)$ at present point (t)
- Traditional Methodology (EKF,UKF, etc.) have a limitation
  - No observation between $t + 1 \sim t + L$
  - Cannot update estimated value efficiently
  - It is hard to predict accurately
- Application of Data-Driven Approach
- Estimate the X_t accurately ⇒ Predict X_{t+L} based on Estimated X_t

β : Ballistic Coefficient

PREVIOUS RESEARCH

Research	Contents	Estimation	Prediction	Prediction with β
Interacting Multiple Model Filter for Tactical Ballistic Missile Tracking [1](2008,AES)	• Propose IMM filter with three kalman filter (Boosting/reentry dynamics , Acceleration , Transition) • Tracking tactical ballistic missiles(TBM)	IMM with KF		
Iterative Joint Integrated Probabilistic Data Association [2] (2013, Fusion)	• Multi target tracking in clutter • Deal with a number of targets in mutual proximity • Propose an iterative implementation of JIPDA • Similar performance with JIPDA but can be used in real time system.	Iterative JIPDA		
Artificial Neural Networks for Estimation and Fusion in Long-Haul Sensor Networks [3] (2015,Fusion)	• Fuse long-haul sensor networks information • Artificial neural network (ANN) • Estimation of ballistic target state	ANN		
Classification and launch-impact point prediction of ballistic target via multiple model maximum likelihood estimator (MM-MLE) [4](2006, IEEE Radar)	• Predict Launch and Impact point of Ballistic Target • Classification the trajectory based on radar measurement data	Estimates burn-out time	• Dynamic equations • Stored parameter	Known β (Stored β in DB)
The novel impact point prediction of a ballistic target with interacting multiple models [5](2013,ICCAS)	• Impact Point Prediction of Ballistic Target • IMM with EKF • Predict the trajectory with 2nd order Runge-Kutta method (calculate the state evolution)	IMM with EKF	2nd order Runge-Kutta	Fixed β (Constant)
Data-Driven Ballistic Coefficient Learning for Future State Prediction of High-Speed Vehicles **(Our model)**	• **Impact Point Prediction(PIP)** of Target • IMM with UKF • Predict target state with predicted beta and dynamic equations	IMM with UKF	• Dynamics equations • **Nonparametric nonlinear Regression** (Gaussian Process)	Unknown β (Predicted β)

PROBLEM DEFINITION

$$Y = \begin{bmatrix} Rang\,e \\ Azim\,t \quad ang\,l \\ Eleva \quad ang \end{bmatrix}$$

High Speed Vehicle Dynamic Model

- ECEF coordinates
- $\dot{\boldsymbol{p}} = \boldsymbol{v}$
  - $\boldsymbol{p} = [x, y, z]^T$ (Position)
- $\dot{\boldsymbol{v}} = a_G + a_D + a_C$
  - $\boldsymbol{v} = [v_x, v_y, v_z]^T$ (Velocity)

$a_G = -\dfrac{\mu p}{||p||^3}$: Gravitational acceleration

$a_D = -\dfrac{\rho}{2\boldsymbol{\beta}}||v||v$: Aerodynamic drag acceleration

$a_C = a_c^{(1)} + a_c^{(2)}$: Coriolis acceleration + Centrifugal acceleration

ρ : Nasa standard atmosphere model

High Speed Vehicle Observation

- Sensor perspective
- $$\boldsymbol{Y} = \begin{bmatrix} \sqrt{(x-x_r)^2 + \text{n}(y-y_r)^2 + (z-z_r)^2} \\ \text{n} \quad \text{ta}^{-1}\left(\frac{y-y_r}{x-x_r}\right) \\ \text{ta}^{-1}\left(\frac{z-z_r}{\sqrt{(x-x_r)^2 + +(y-y_r)^2}}\right) \end{bmatrix} + e$$

(x_r, y_r, z_r) : Sensor position

e : Measurement noise

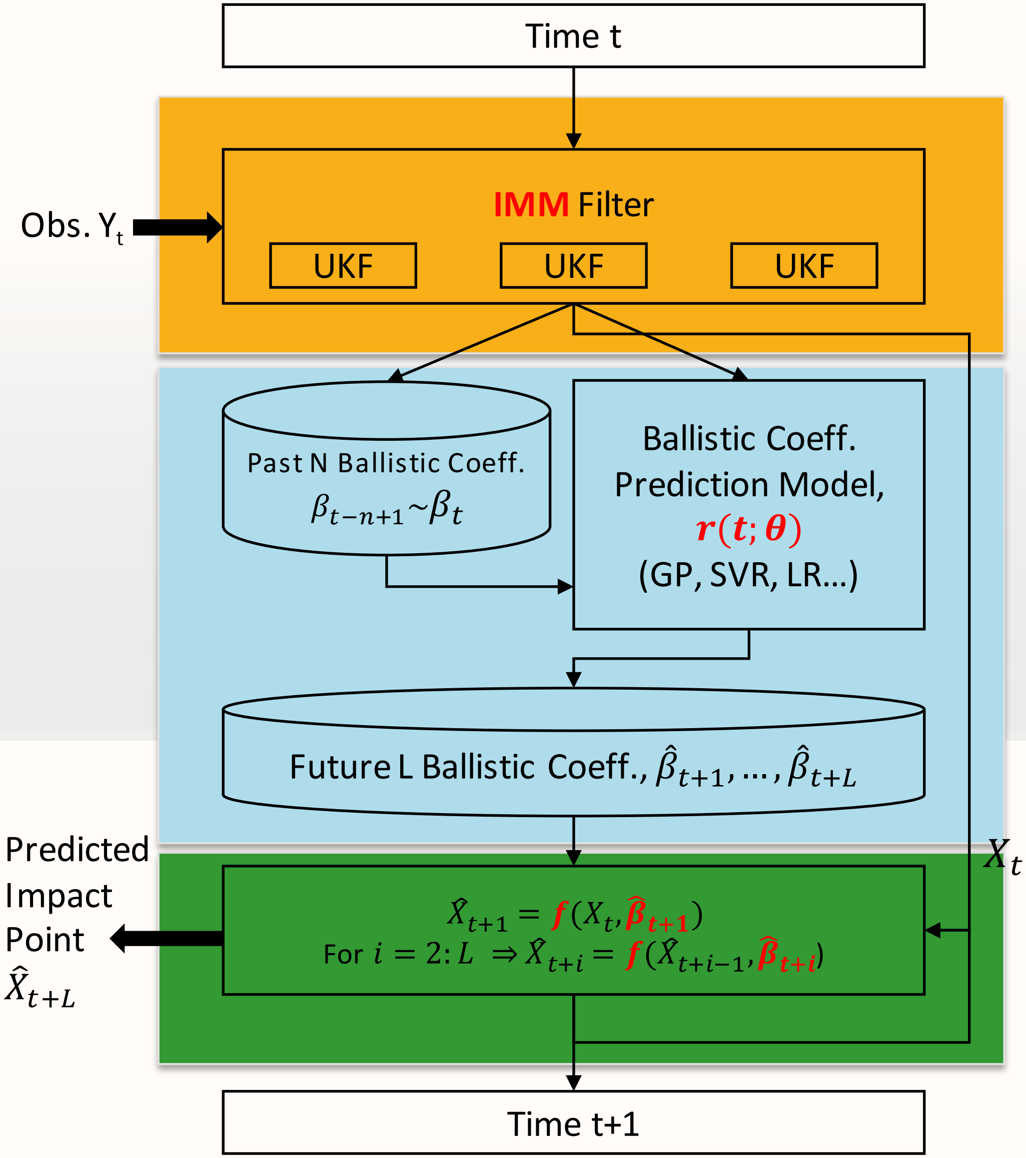
Ballistic coefficient

- Ballistic coefficient β is unknown parameter
- Accurate estimation of β is needed
  - β is used to calculate aerodynamic drag acceleration
- Estimates β using IMM(Interacting Multiple Model)
- Predict future β with several regression models

METHODOLOGY–OVERVIEW

Current Time : t
Lookahead Time : L (Predict $\hat{X}_{t+L}$ at time t)
$r(t;\theta)$: regression function
- θ : hyperparameter of regression function

f : Dynamic Equations



- Utilize Observation Data
- **IMM with UKF**
- $\beta_1 \sim \beta_t$, $X_1 \sim X_t$ Estimation

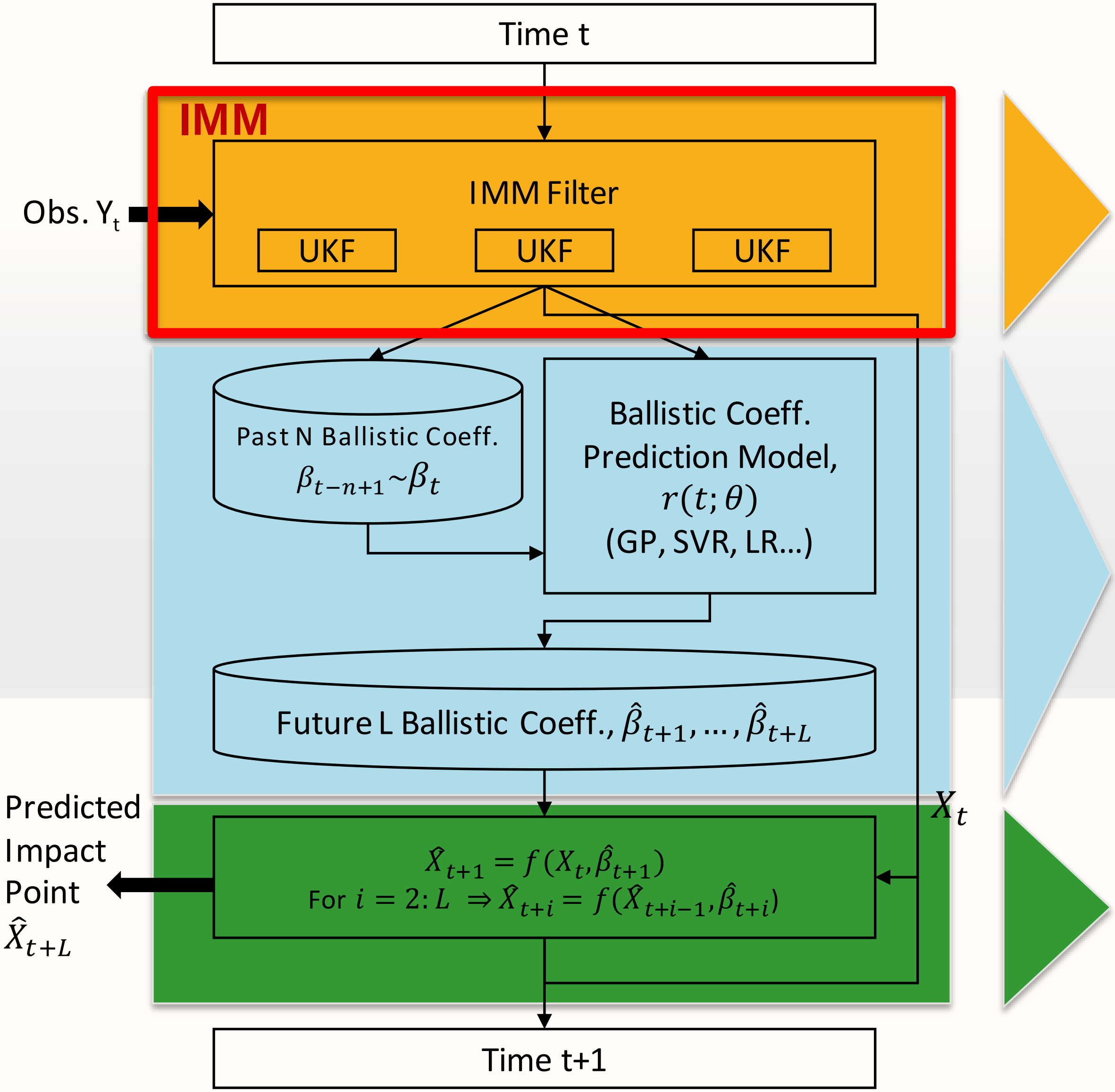
- Use estimated β drawn by IMM $(\beta_{t-n+1}, \dots, \beta_t)$
- Predict $\hat{\beta}_{t+1} \sim \hat{\beta}_{t+L}$ based on $\beta_{t-n+1} \sim \beta_t$
- Utilize several kinds of **regression** methods
  - None (constant)
  - Regularized Linear Regression
  - Support Vector Regression
  - Gaussian Process Regression

- **Predict $\hat{X}_{t+L}$** based on $\hat{\beta}_{t+1} \sim \hat{\beta}_{t+L}$, $\hat{X}_{t+1}, \dots, \hat{X}_{t+L-1}$
- Use dynamics Equation recursively

METHODOLOGY-OVERVIEW

Current Time : t
Lookahead Time : L (Predict $\hat{X}_{t+L}$ at time t)
$r(t;\theta)$: regression function
- θ : hyperparameter of regression function

f : Dynamic Equations



- Utilize Observation Data
- IMM with UKF
- $\beta_1\sim\beta_t$, $X_1\sim X_t$ Estimation

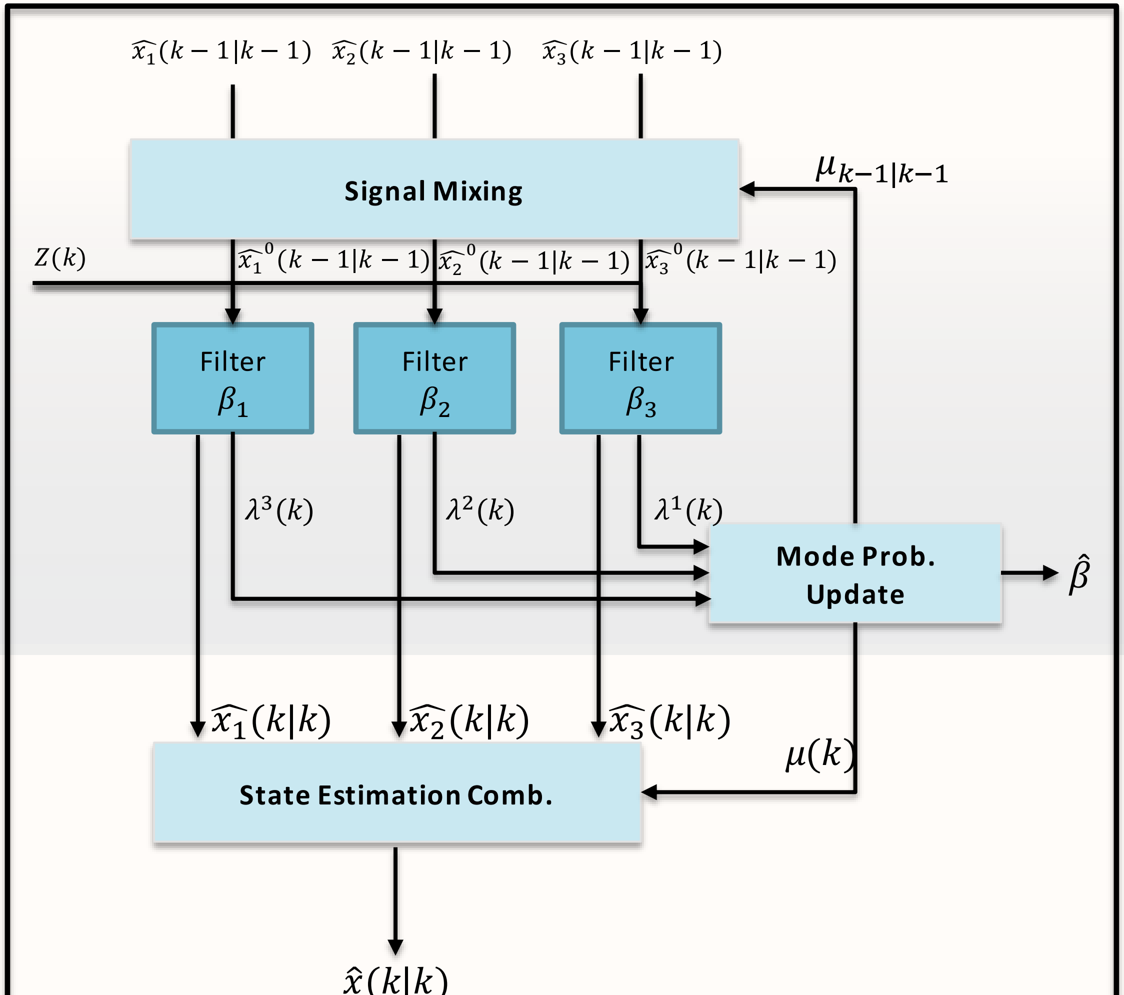
- Use estimated β drawn by IMM $(\beta_{t-n+1},\dots,\beta_t)$
- Predict $\hat{\beta}_{t+1}\sim\hat{\beta}_{t+L}$ based on $\beta_{t-n+1}\sim\beta_t$
- Utilize several kinds of regression methods
  - None (constant)
  - Regularized Linear Regression
  - Support Vector Regression
  - Gaussian Process Regression

- Predict $\hat{X}_{t+L}$ based on $\hat{\beta}_{t+1}\sim\hat{\beta}_{t+L}$, $\hat{X}_{t+1},\dots,\hat{X}_{t+L-1}$
- Use dynamics Equation recursively

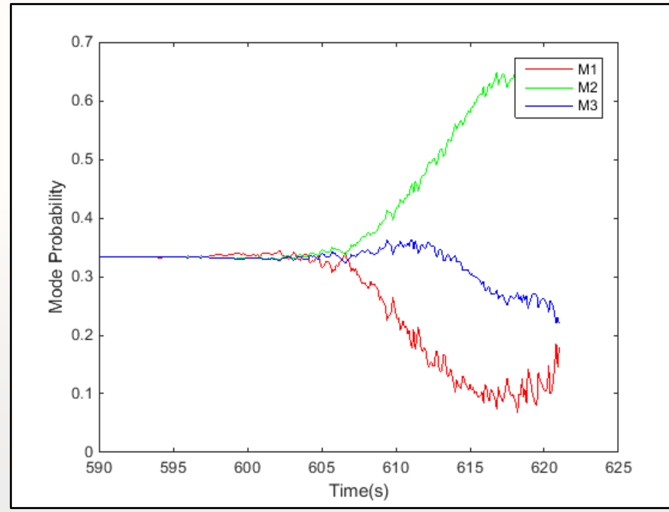
METHODOLOGY-IMM

Mode = [M1,M2,M3] = [10000, 110000, 210000]

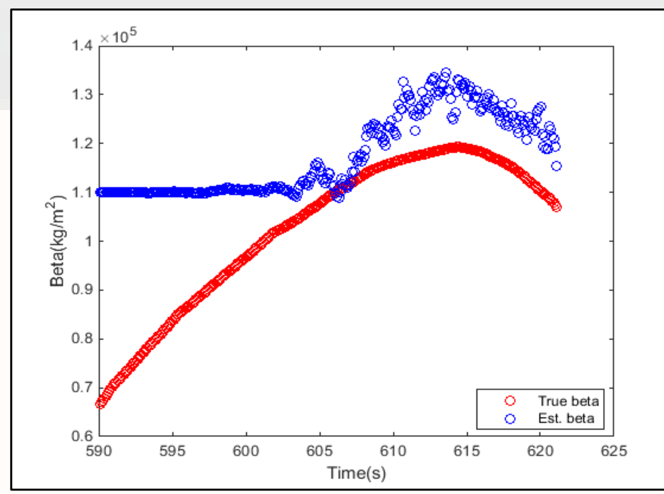
IMM (Interacting Multiple Model) [6]



Mode Probability



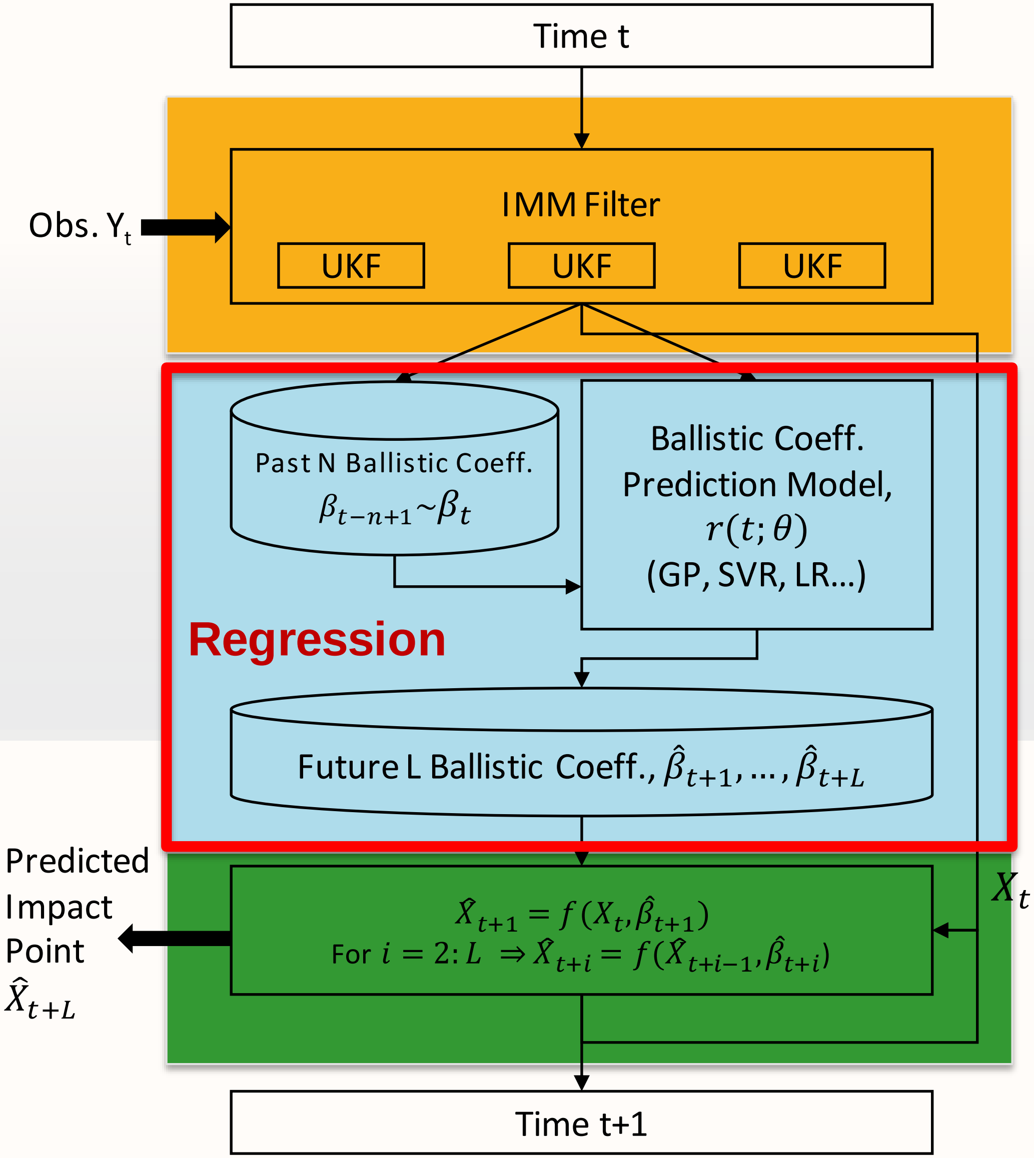
Beta Estimation



METHODOLOGY-OVERVIEW

Current Time : t
Lookahead Time : L (Predict $\hat{X}_{t+L}$ at time t)
$r(t;\theta)$: regression function
- θ : hyperparameter of regression function

f : Dynamic Equations

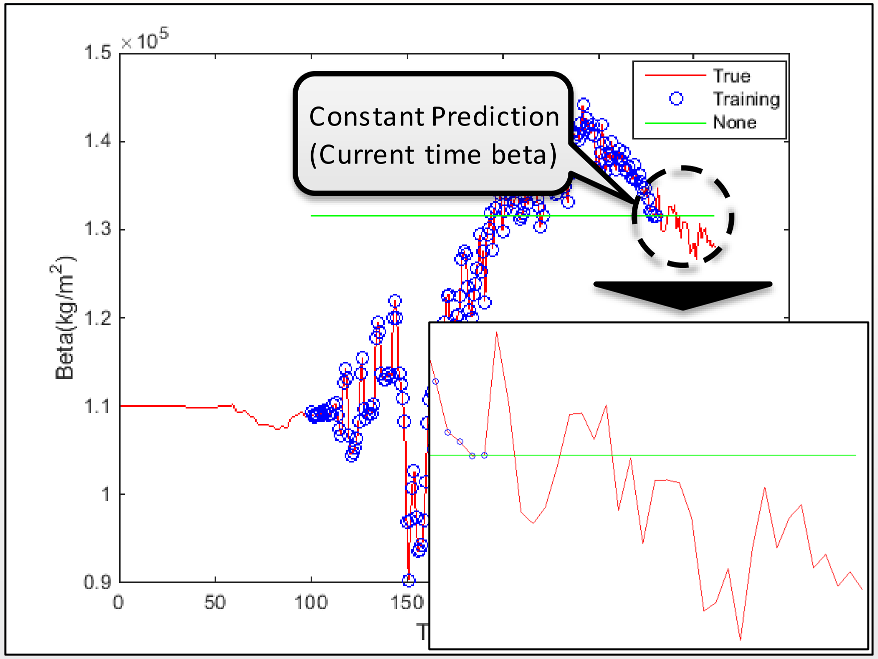


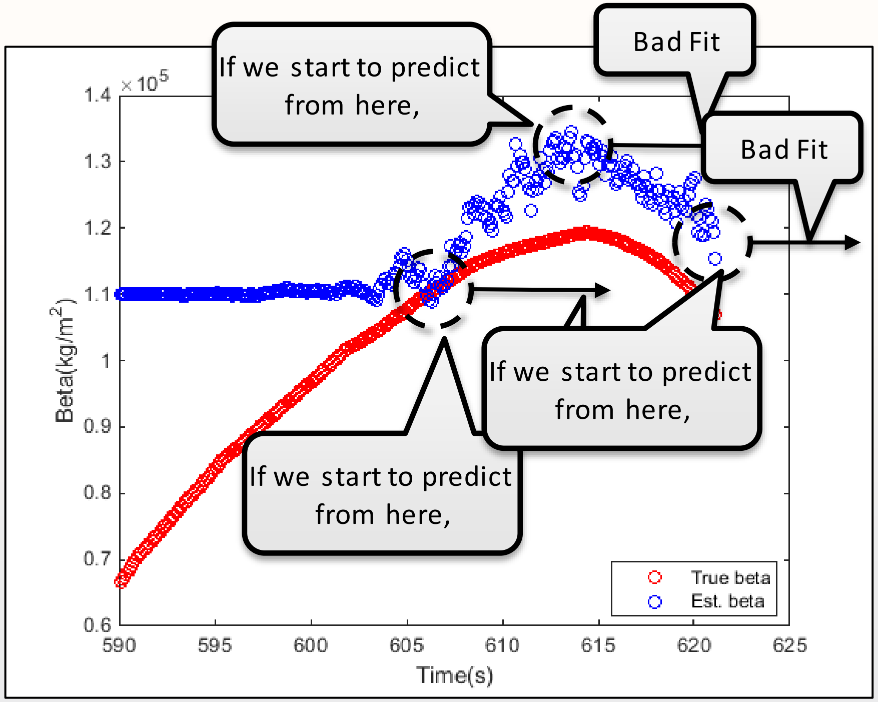
- Utilize Observation Data
- IMM with UKF
- $\beta_1 \sim \beta_t$, $X_1 \sim X_t$ Estimation

- Use estimated β drawn by IMM $(\beta_{t-n+1}, \ldots, \beta_t)$
- Predict $\hat{\beta}_{t+1} \sim \hat{\beta}_{t+L}$ based on $\beta_{t-n+1} \sim \beta_t$
- Utilize several kinds of regression methods
  - **None (constant)**
  - **Regularized Linear Regression**
  - **Support Vector Regression**
  - **Gaussian Process Regression**
    - **With various kernels.**

- Predict $\hat{X}_{t+L}$ based on $\hat{\beta}_{t+1} \sim \hat{\beta}_{t+L}$, $\hat{X}_{t+1}, \ldots, \hat{X}_{t+L-1}$
- Use dynamics Equation recursively

METHODOLOGY – NONE (CONSTANT)





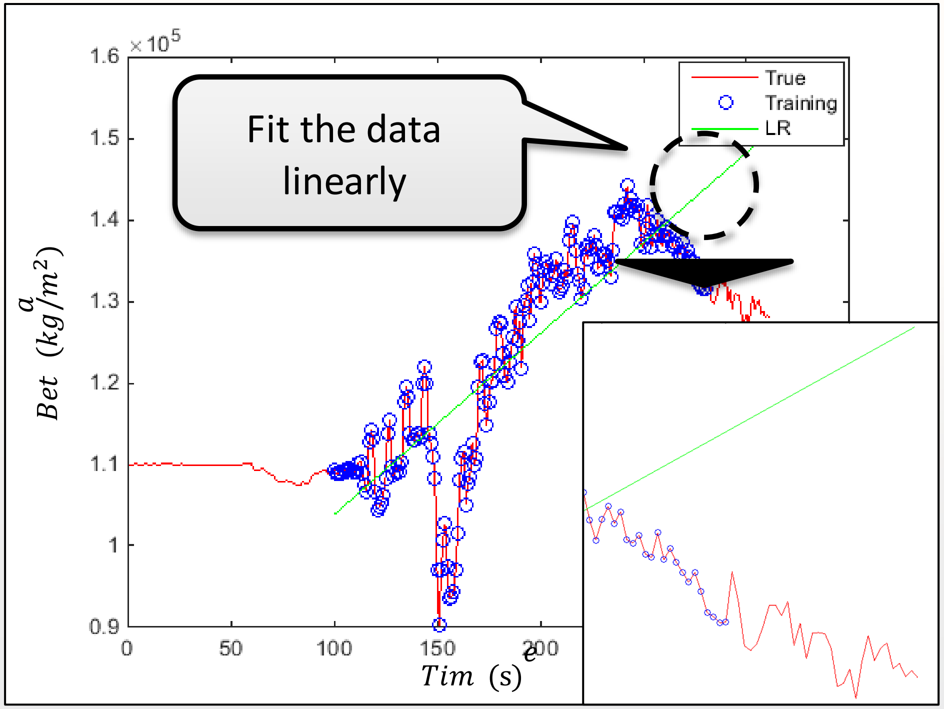
<u>None (Constant Prediction)</u>

- Use a current beta value as a predicted beta value
  - (Last estimated beta)
- Assume that $(\beta_1, \dots, \beta_t)$ are estimated $\Rightarrow \left(\widehat{\beta_{t+1}}, \dots, \widehat{\beta_{t+L}}\right) = (\beta_t, \dots, \beta_t)$
- Traditional Approach

Nonlinearity of beta

- True beta and Estimated beta have nonlinearity form (similar with quadratic)
- None(Constant regression) is hard to fit beta well

METHODOLOGY – REGULARIZED LR



Overfitting

At the symphony...

- Want to listen to symphony only
  - Without noise
  - ex) neighbors shuffling
- Overfitting
  - Hear more noise than we need

- William Chen, Data Scientist at Quora

At the target tracking…

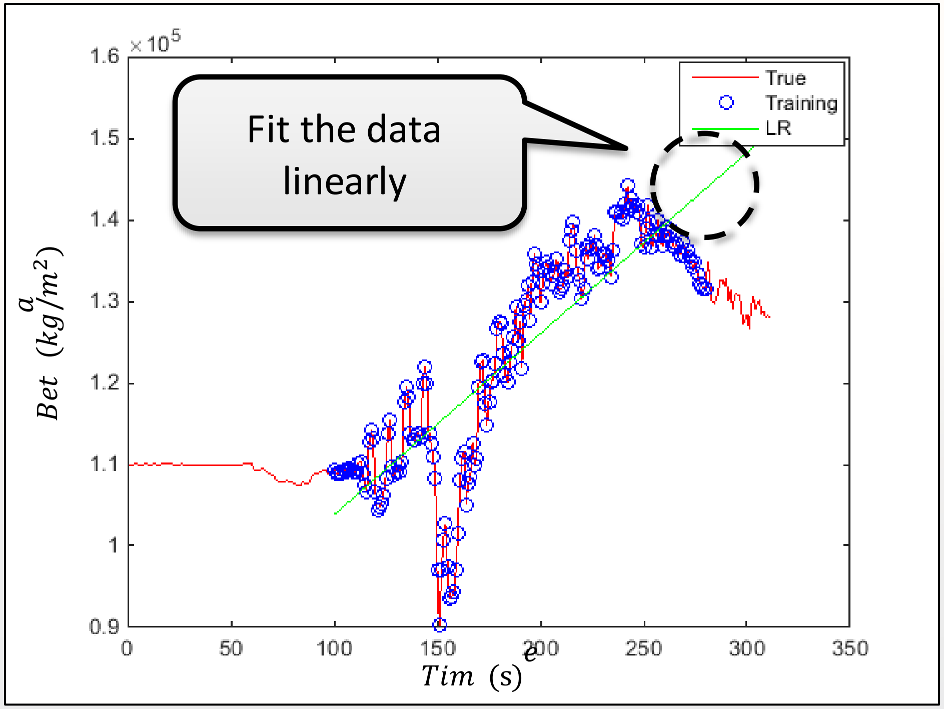
- **Many observation noises**
- Want to identify signal and noise
- Model as much signal as possible
- Model as little noise as possible

Regularized Linear Regression [7]

- $L2$ Regularization
- $E(\theta) = \frac{1}{2}\sum_{j=0}^{j=N}\{(\beta_{t-j} - y(T_j,\theta)\}^2 + \frac{\lambda}{2}||\boldsymbol{\theta}||^2$
- $\theta_t^{LR,*} = (T^T T + \lambda I)^{-1} T^T \cdot B$
  - T : Collection of Time Points
    - $< t-N, \; \ldots, \; t >$
  - $\bar{B}$: Collection of ballistic coefficient estimates
    - $< \beta_{t-N}, \ldots, \beta_t >$

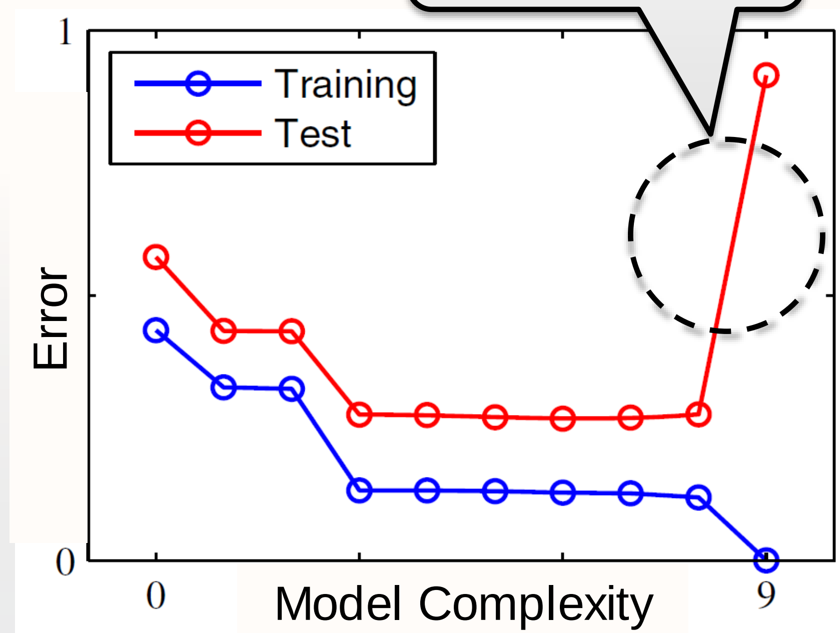
- Ridge Regression
- More Stable than least squares (normal linear regression)
- Prevent **Overfitting**
  - Prevent sum of norms of the slopes get too high

METHODOLOGY – REGULARIZED LR



Training error becomes small
Test error becomes very large

Overfitting[8]



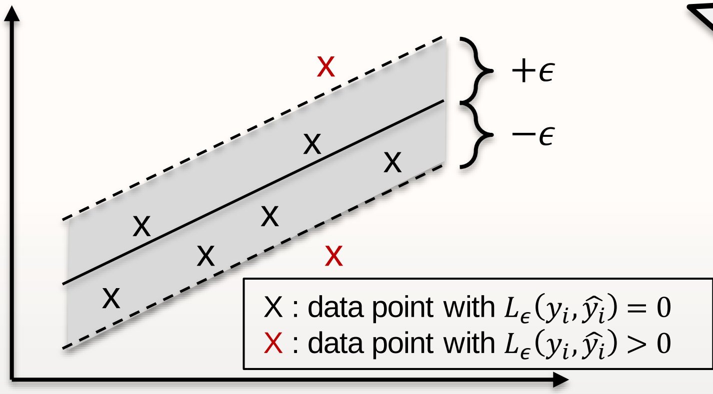
Regularized Linear Regression [7]

- $L2$ Regularization
- $E(\theta) = \frac{1}{2}\sum_{j=0}^{j=N}\{(\beta_{t-j} - y(T_j,\theta)\}^2 + \frac{\lambda}{2}||\boldsymbol{\theta}||^2$
- $\theta_t^{LR,*} = (T^T T + \lambda I)^{-1} T^T \cdot B$
  - T : Collection of Time Points
    - $< t-N, \ \dots\ , \ t >$
  - $\bar{B}$: Collection of ballistic coefficient estimates
    - $< \beta_{t-N}, \dots, \beta_t >$

- Ridge Regression
- More Stable than least squares (normal linear regression)
- Prevent **Overfitting**
  - Prevent sum of norms of the slopes get too high

METHODOLOGY – SVR

Support Vector Regression [9,10]



- Nonlinear Regression with kernel
- **Hard to optimize kernel hyper parameter**
- Cross validation also has a limitation to optimization
  - Parameters only depend on human's choice
  - Ex) if $\lambda_{candidate} = [0.1, 0.5, 1, 2, 10]$, λ should be one of $\lambda_{candidate}$

$$L_\epsilon(y_i, \hat{y}_i) = \begin{cases} 0 & \text{If } |y_i, \hat{y}_i| < \epsilon \\ |y_i, \hat{y}_i| - \epsilon & \text{Otherwise} \end{cases}$$

$$\min \; C \sum_{i=1}^{N} \boldsymbol{L_\epsilon(y_i, \hat{y}_i)} + \frac{1}{2}||w||^2$$

$$s.t.\ \hat{y}_i = f(x_i) = w^T x_i + w_0$$

Kernel (Non-linear)

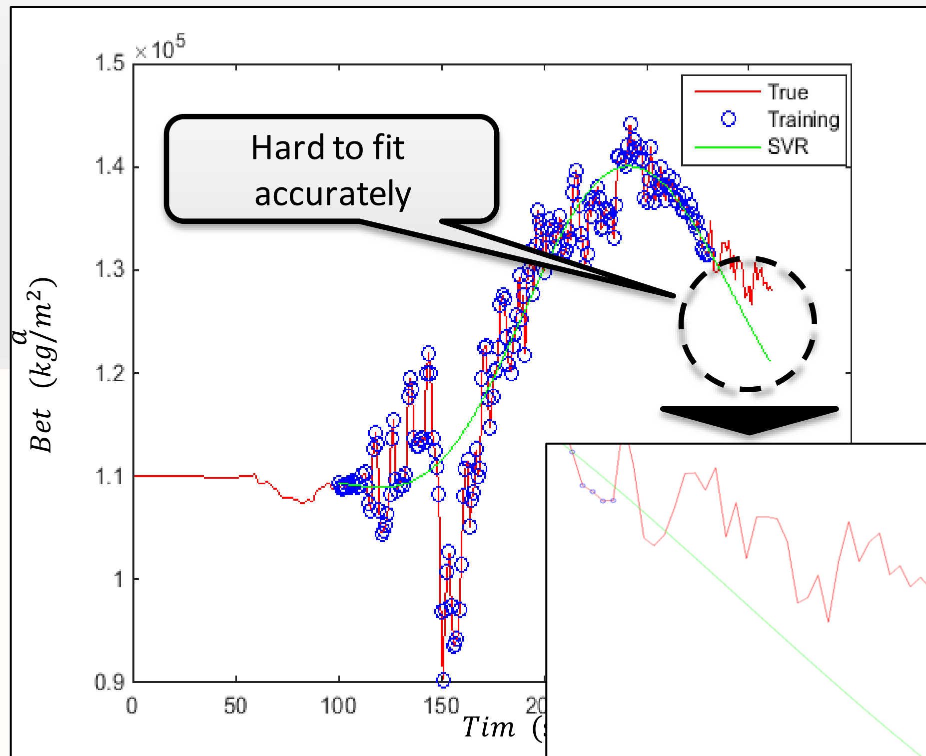
Dual form

$$\max \begin{cases} -\frac{1}{2} \sum_{i,j=1}^{N} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) \boldsymbol{k(T_i, T_j)} \\ -\epsilon \sum_{i=1}^{N} (\alpha_i + \alpha_i^*) + \sum_{i=1}^{N} \beta_i (\alpha_i - \alpha_i^*) \end{cases}$$

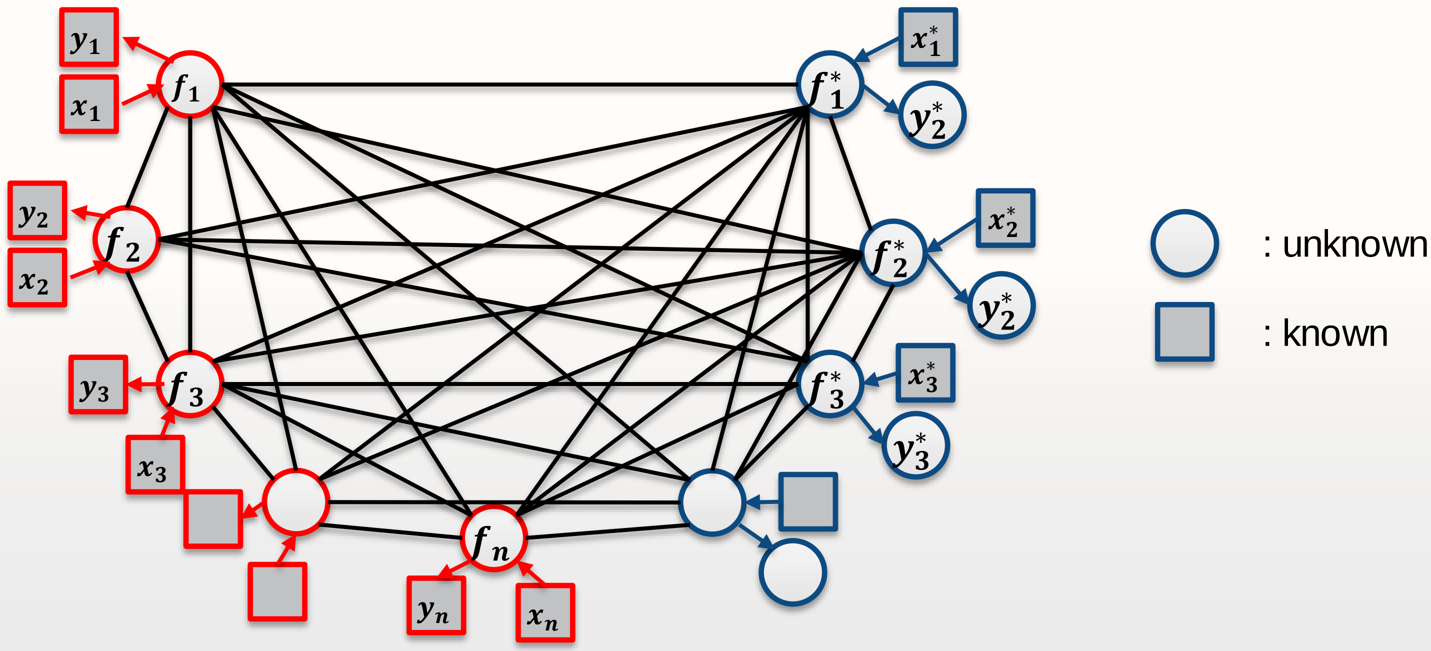
$$\text{such that } \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) = 0$$

$$\text{where } \alpha_i, \alpha_i^* \in [0, C]$$

Example of SVR



METHODOLOGY – GP INTRO I



GP (Gaussian Process) is a stochastic process such that any finite sub collection of random variables has a multivariate Gaussian Distribution **[11]**

$$\Rightarrow \boldsymbol{f \sim GP(m(x), k(x, x'))}$$

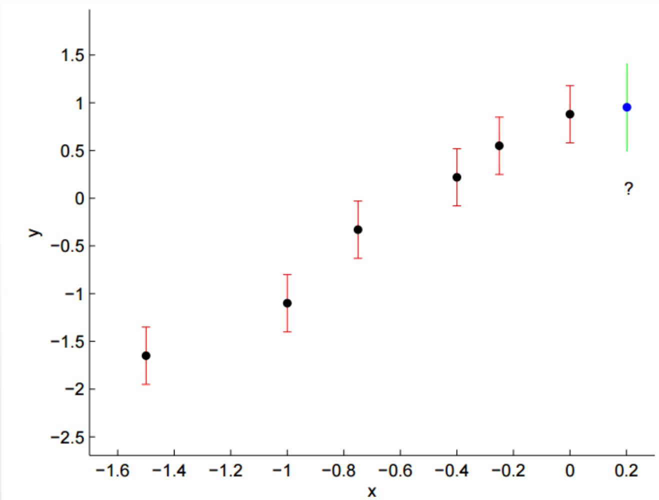
Joint of Gaussian distribution is also Gaussian $\Rightarrow \begin{bmatrix} f \\ f^* \end{bmatrix} \sim \boldsymbol{N}\left(\begin{bmatrix} \mu \\ \mu^* \end{bmatrix}, \begin{bmatrix} K & K_* \\ K_*^T & K_{**} \end{bmatrix}\right)$

Conditional Gaussian distribution is also Gaussian $\Rightarrow$ $f_* | X_*, X, f \sim \boldsymbol{N}(f_* | \mu_*, \Sigma_*)$

$\mu_* = \mu(X_*) + K_*^T K^{-1}(f - \mu(X))$
$\Sigma_* = K_{**} - K_*^T K^{-1} K_*$
whe $K = k(x,x)$, $K_* = k(x, x_*)$, $K_{**} = k(x_*, x_*)$

Stochastic Process is a collection of random variables $\{f(x) : x \in X\}$

METHODOLOGY – GPR INTRO I

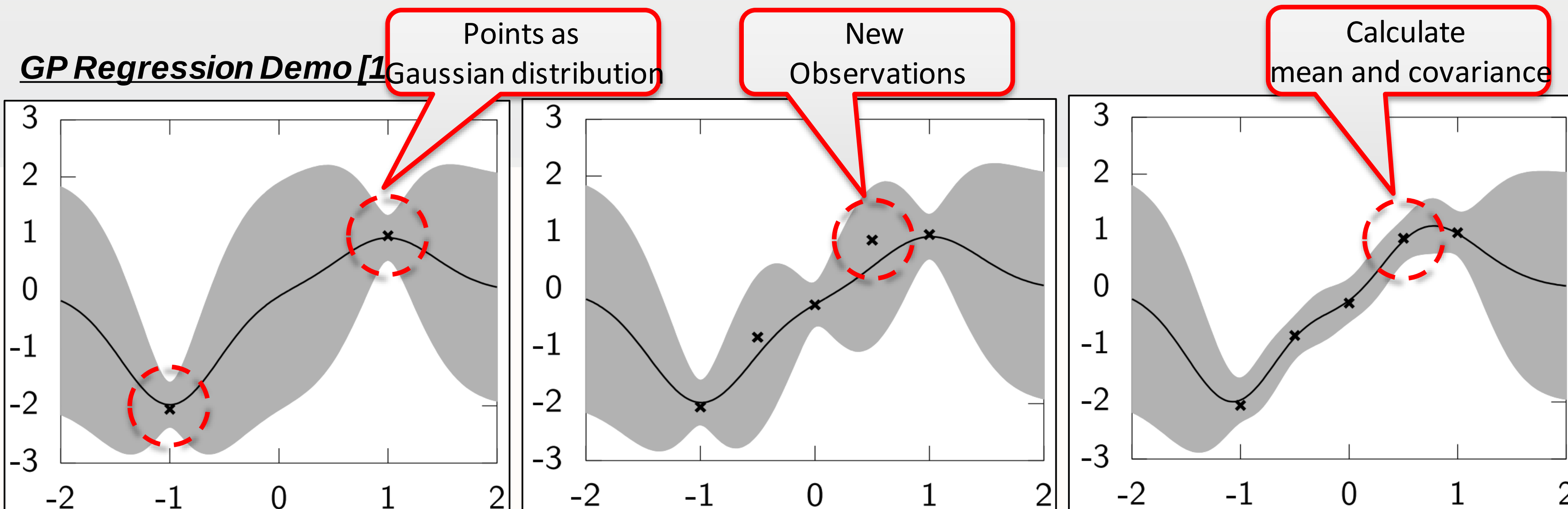


Consider a point as Gaussian distribution **[12]**
⇒ A point has its mean and sigma individually

When a new point is observed
⇒ Consider it as a normal distribution also
⇒ Can predict its mean and sigma
$(\because) Condit\ i\ of\ Guass\ i\ is\ als\ o\ Gauss\ i$

$$\boldsymbol{f_* | X_*, X, f \sim N(f_* | \mu_*, \Sigma_*)}$$

GP Regression Demo [1



METHODOLOGY – GPR INTRO II

Gaussian Process Regression [14]

- $\beta_i = f(T_i) + \epsilon_i \;, (i = t - N, \dots, t)$
  - Noise : $\epsilon_i \sim N(0, \sigma^2)$
  - Prior : $f(\cdot) \sim GP(0, K(\cdot,\cdot))$
- $\log P(\beta | T, \theta) = -\frac{1}{2} \log |K^*| - \frac{1}{2} \beta^T (K^*)^{-1} \beta - \frac{N+1}{2} \log 2$
  - $K^* = K(T, T) + \sigma^2 I$
- $\frac{\partial \log P(\beta | T, \theta)}{\partial \theta} = -\frac{1}{2} \mathrm{tra}\left(K^{*-1} \frac{\partial K^*}{\partial \theta}\right) + \frac{1}{2} \beta^T K^{*-1} \frac{\partial K^*}{\partial \theta} \beta$
- $\widehat{\beta_{t+L}} = K(T_*, T) {K^*}^{-1} \beta$
  - T_* : $t + 1, \dots, t + L$ (Test Point)

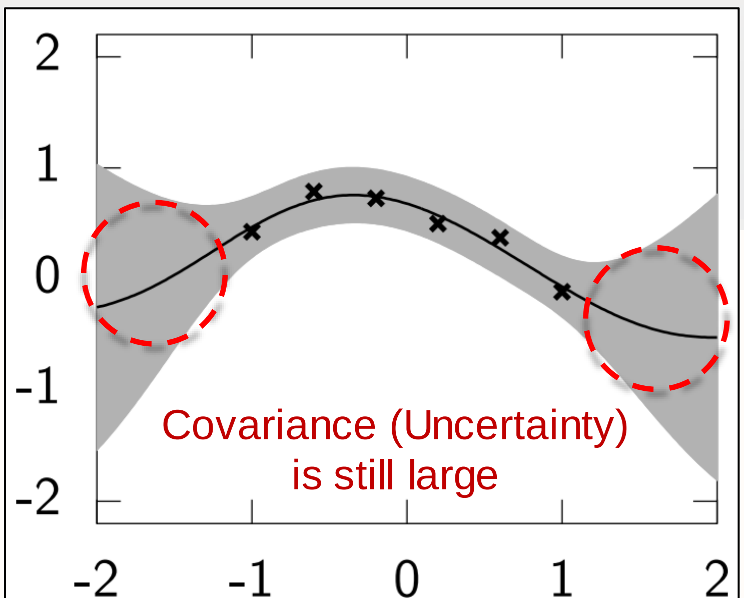
GPR is Nonparametric nonlinear regression
Possible to optimize hyper parameter
- θ is a parameter of K (kernel)
- Find θ which maximize likelihood

Performs better than SVR usually

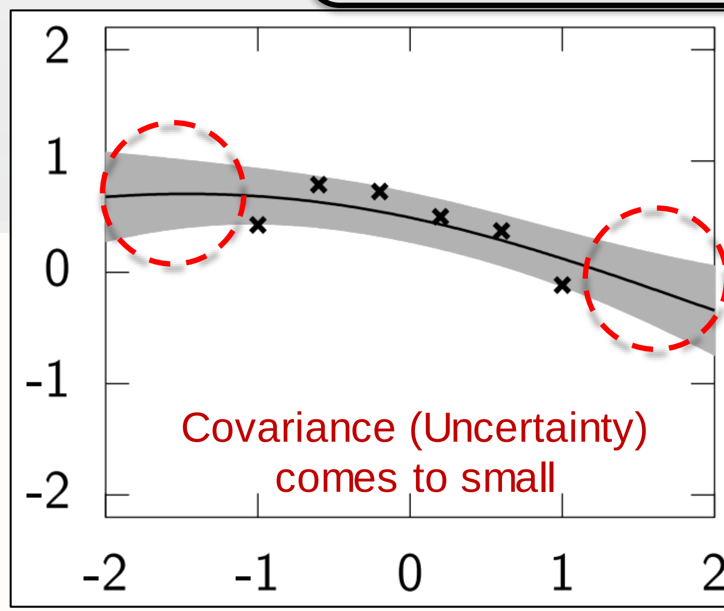
Example

$K(T_i, T_j) = \sigma_p^2 \exp\left(-\frac{(T_i - T_j)^2}{2 l^2}\right)$

$\Rightarrow \theta = \sigma_p^2 \;, l$



GPR without optimization [13]



GPR with optimization [13]

METHODOLOGY – GPR AND SVR

Gaussian Process Regression [14]

- $\beta_i = f(T_i) + \epsilon_i, (i = t - N, \dots, t)$
  - Noise : $\epsilon_i \sim N(0, \sigma^2)$
  - Prior : $f(\cdot) \sim GP(0, K(\cdot,\cdot))$
- $\mathrm{lo}\ P(\beta|T,\theta) = -\frac{1}{2}\mathrm{lo}\ |K^*| - \frac{1}{2}\beta^T (K^*)^{-1}\beta - \frac{N+1}{2} log$ 2
  - $K^* = K(T,T) + \sigma^2 I$
- $\frac{\partial \mathrm{lo}\ P(\beta|T,\theta)}{\partial\theta} = -\frac{1}{2} tra \left(K^{*-1}\frac{\partial K^*}{\partial\theta}\right) + \frac{1}{2}\beta^T K^{*-1}\frac{\partial K^*}{\partial\theta}\beta$
- $\widehat{\beta_{t+L}} = K(T_*, T) K^{*-1}\beta$
  - T_* : $t+1, \dots, t+L$ (Test Point)

GPR is Nonparametric nonlinear regression

Possible to optimize hyper parameter

- θ is a parameter of K (kernel)
- Find θ which maximize likelihood

Performs better than SVR usually

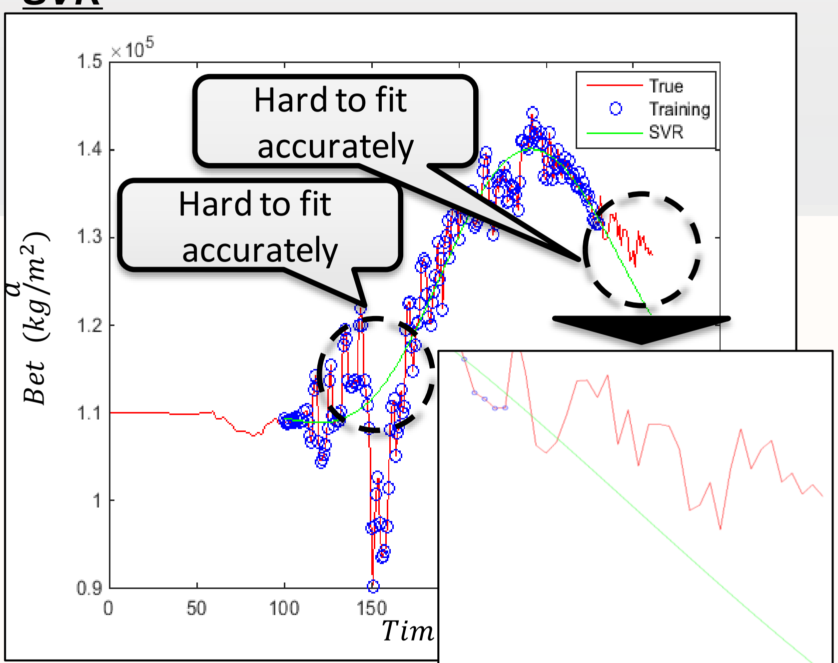
- **SVR : Hard to optimize hyper parameter**

Example

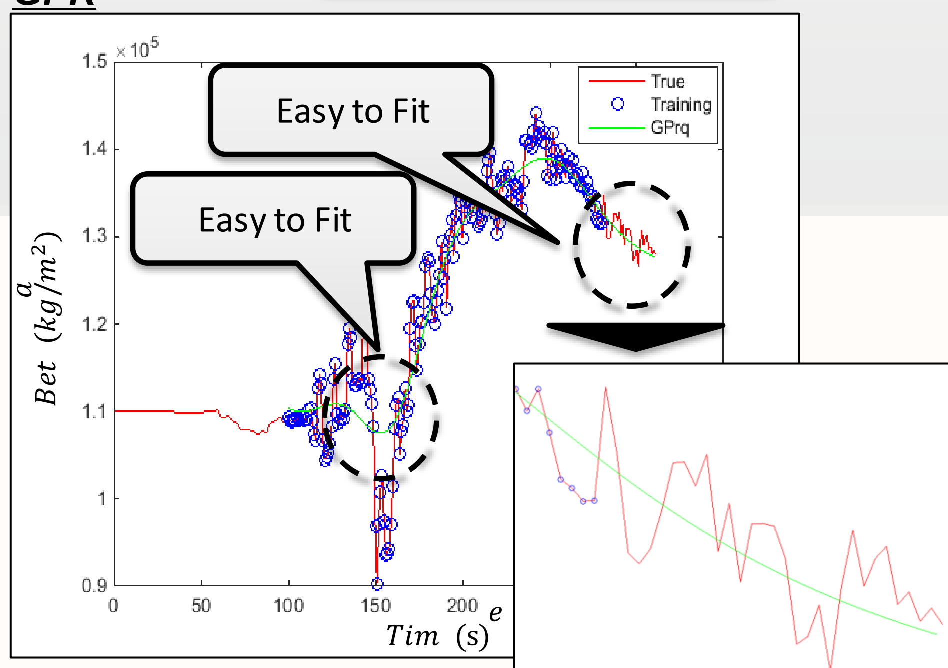
$$K(T_i, T_j) = \sigma_p^2 \mathrm{ex}\ (-\frac{(T_i - T_j)^2}{2l^2})$$

$$\Rightarrow \theta = \sigma_p^2, l$$

SVR



GPR



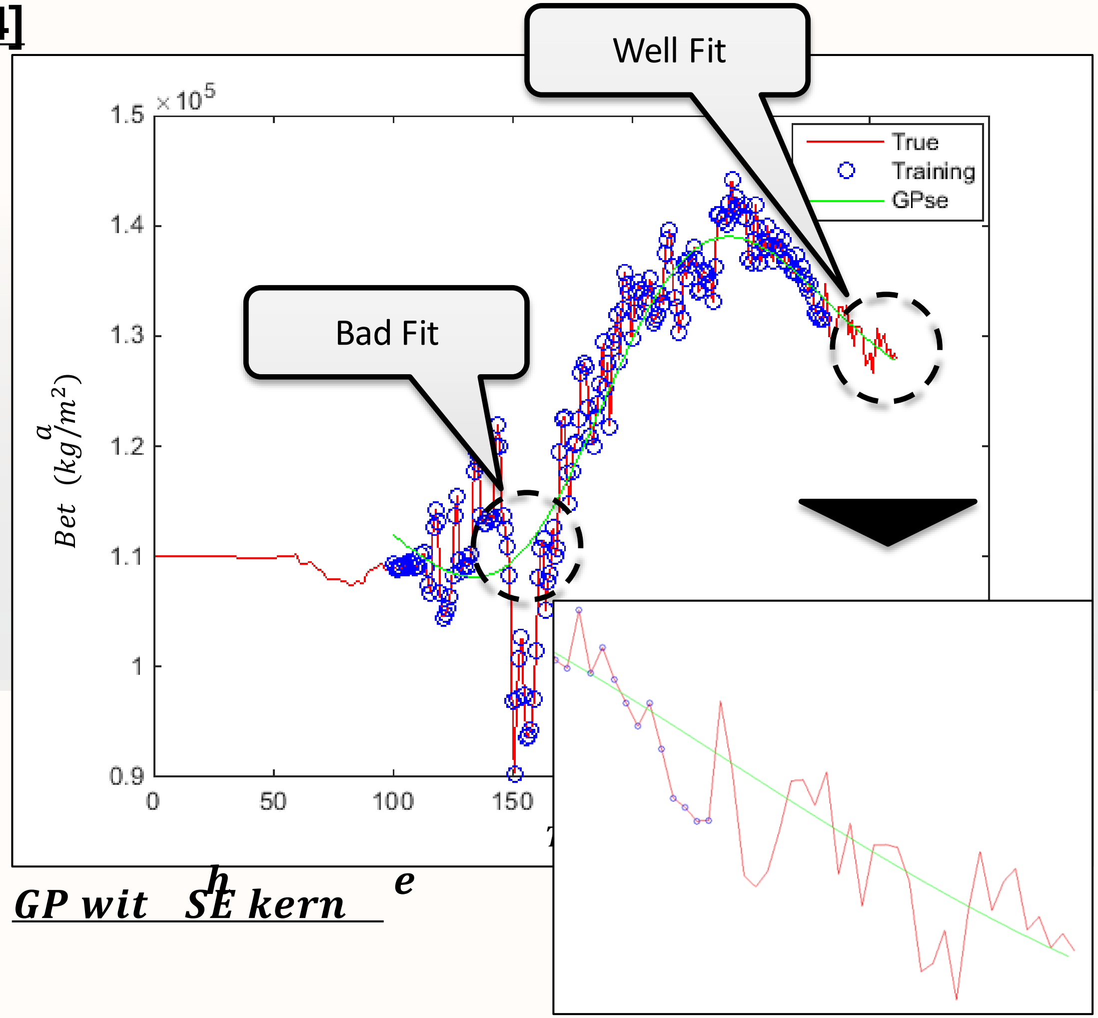
METHODOLOGY – GP SE

Gaussian Process Regression [14]

- Nonparametric Nonlinear Regression
  - Kernel Calculation
- $\beta_i = f(T_i) + \epsilon_i\ ,(i = t-N, \dots, t)$
  - Noise : $\epsilon_i \sim N(0,\sigma^2)$
  - Prior : $f(\cdot) \sim GP(0, \boldsymbol{K}(\cdot,\cdot))$
- $\log P(\beta|T,\theta)$
  $= -\frac{1}{2}\log|K^*| - \frac{1}{2}\beta^T(K^*)^{-1}\beta - \frac{N+1}{2}log\ 2$
  - $K^* = K(T,T) + \sigma^2 I$
- $\frac{\partial \log P(\beta|T,\theta)}{\partial\theta}$
  $= -\frac{1}{2} tra\left(K^{*-1}\frac{\partial K^*}{\partial\theta}\right) + \frac{1}{2}\beta^T K^{*-1}\frac{\partial K^*}{\partial\theta}\beta$
- $\widehat{\beta_{t+L}} = K(T_*,T)K^{*-1}\beta$
  - $T_* : t+1, \dots, t+L$ (Test Point)

Kernel [14]

- Kind of similarity
  - Between data points
- **SE**(Squared Exponential) Kernel
  - $k_{SE}(T_i,T_j) = \sigma_p^2 \exp\left(-\frac{(T_i-T_j)^2}{2l^2}\right)$
  - σ_p^2 : determine average distance
  - l : determine wiggle



GP with SE kernel

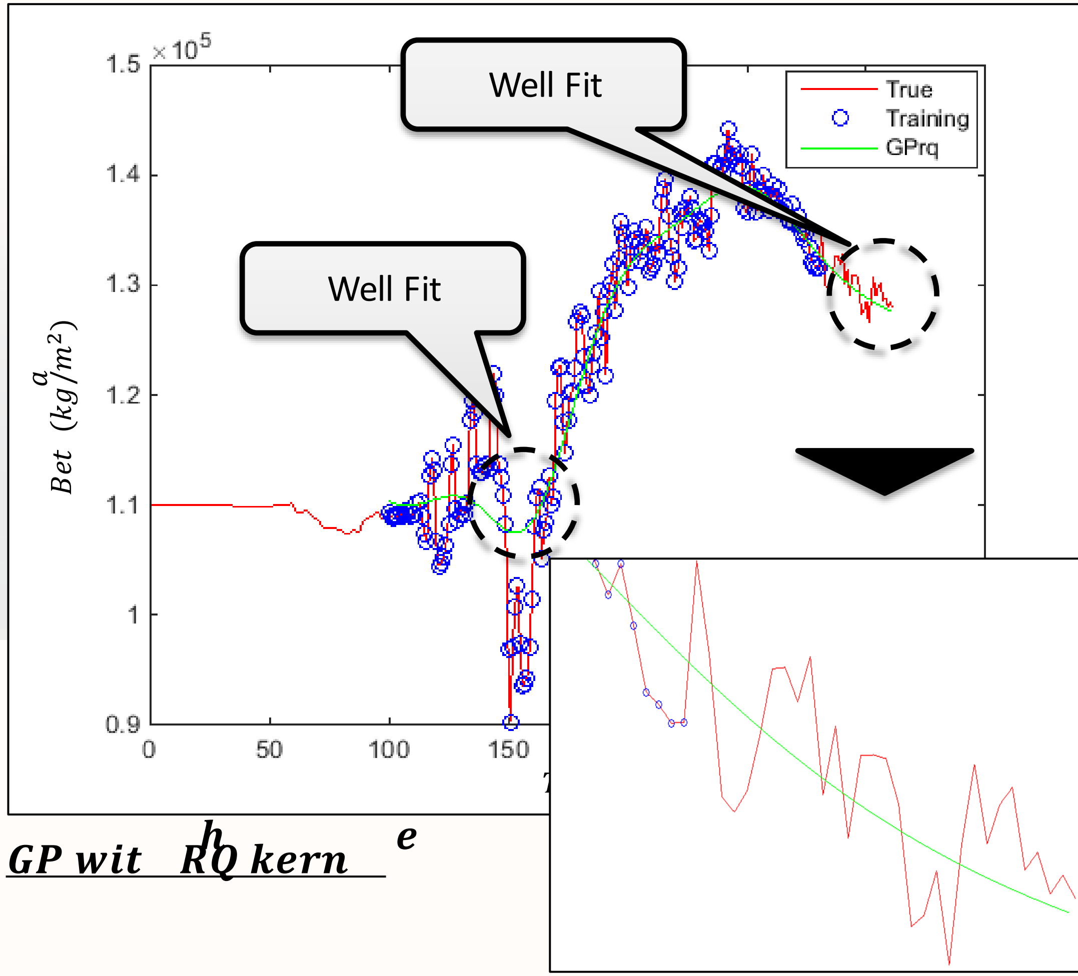
METHODOLOGY – GP RQ

Gaussian Process Regression [14]

- Nonparametric Nonlinear Regression
  - Kernel Calculation
- $\beta_i = f(T_i) + \epsilon_i \,, (i = t-N, \ldots, t)$
  - Noise : $\epsilon_i \sim N(0, \sigma^2)$
  - Prior : $f(\cdot) \sim GP(0, \boldsymbol{K}(\cdot,\cdot))$
- $\log P(\beta|T,\theta)$
  $= -\frac{1}{2}\log|K^*| - \frac{1}{2}\beta^T (K^*)^{-1}\beta - \frac{N+1}{2}\log 2$
  - $K^* = K(T,T) + \sigma^2 I$
- $\frac{\partial \log P(\beta|T,\theta)}{\partial \theta}$
  $= -\frac{1}{2} trace\left(K^{*-1}\frac{\partial K^*}{\partial \theta}\right) + \frac{1}{2}\beta^T K^{*-1}\frac{\partial K^*}{\partial \theta}\beta$
- $\widehat{\beta_{t+L}} = K(T_*, T) {K^*}^{-1} \beta$
  - T_* : $t+1, \ldots, t+L$ (Test Point)

Kernel [14]

- Kind of similarity
  - Between data points
- **RQ**(Rational Quadratic) Kernel
  - $k_{RQ}(T_i, T_j) = \sigma_p^2 \left(1 + \frac{(T_i - T_j)^2}{2\alpha l^2}\right)^{-\alpha}$



GP with RQ kernel

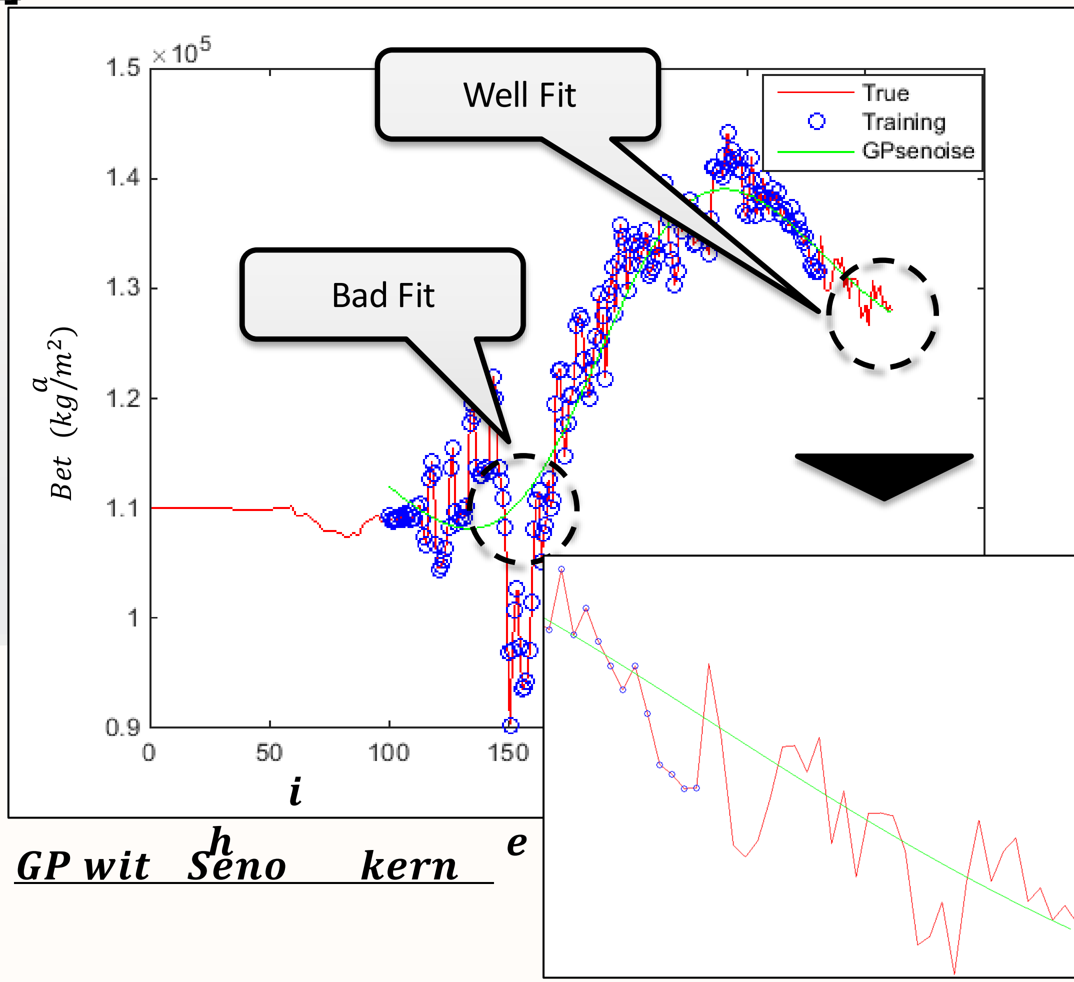
METHODOLOGY – GP SENOISE

Gaussian Process Regression [14]

- Nonparametric Nonlinear Regression
  - Kernel Calculation
- $\beta_i = f(T_i) + \epsilon_i \, , (i = t - N, \ldots, t)$
  - Noise : $\epsilon_i \sim N(0, \sigma^2)$
  - Prior : $f(\cdot) \sim GP(0, \boldsymbol{K}(\cdot,\cdot))$
- $\log P(\beta | T, \theta)$
  $= -\frac{1}{2}\log|K^*| - \frac{1}{2}\beta^T (K^*)^{-1}\beta - \frac{N+1}{2} \log 2$
  - $K^* = K(T,T) + \sigma^2 I$
- $\frac{\partial \log P(\beta|T,\theta)}{\partial\theta}$
  $= -\frac{1}{2} trace\left(K^{*-1}\frac{\partial K^*}{\partial\theta}\right) + \frac{1}{2}\beta^T K^{*-1}\frac{\partial K^*}{\partial\theta}\beta$
- $\widehat{\beta_{t+L}} = K(T_*, T)K^{*-1}\beta$
  - $T_* : t+1, \ldots, t+L$ (Test Point)

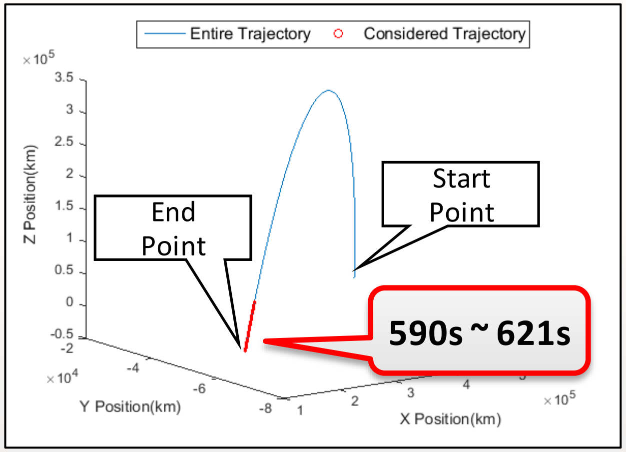
Kernel [14]

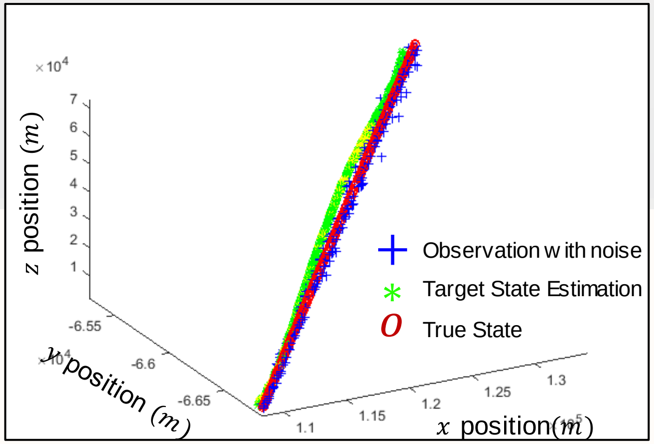
- Kind of similarity
  - Between data points
- **SEnoise**(Squared Exponential Noise Kernel)
  - $k_{SENoise}(T_i, T_j) = \sigma_p^2 \exp\left(-\frac{(T_i - T_j)^2}{2l^2}\right) + w_{noise}\,\delta_{T_i,T_j}$



GP with SEnoise kernel

METHODOLOGY – EXPERIMENTAL SETTING

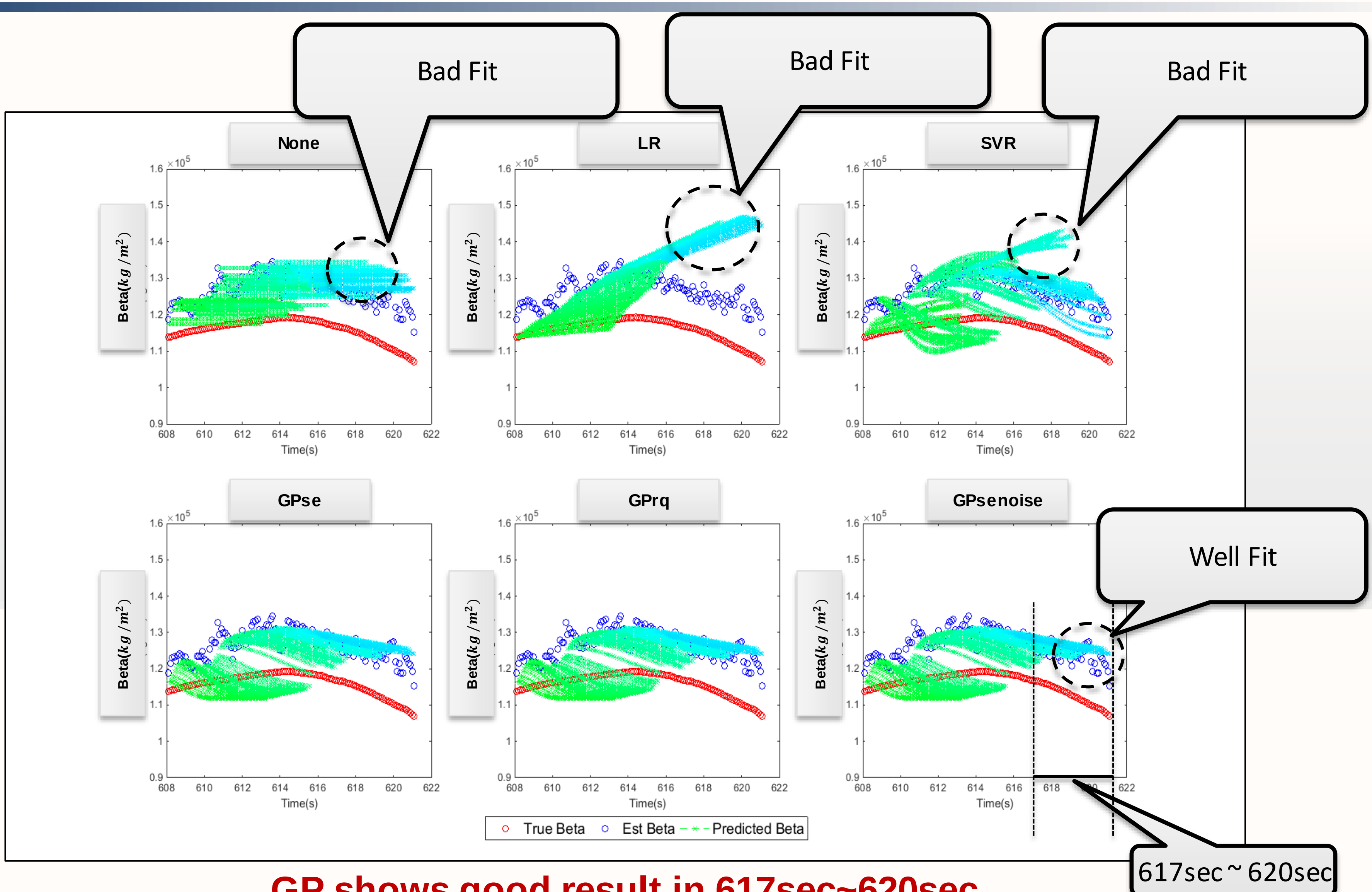




High-Speed vehicle flight time: 0s~621S
Sensor detection time : 590s~621s

- **Estimate Beta and Position using UKF-IMM**
  - Beta Hypothesis = [10000 110000 210000]
- **Predict Beta $\boldsymbol{\beta_{t+1} \sim \beta_{t+L}}$, using Regression**
  - TrainingBeta = [5s , 10s, 15s]
    - The number of recent beta used to training regression model
    - If we predict $\beta_{t+1} \sim \beta_{t+L}$ based on $\beta_{t-n+1} \sim \beta_t$,
    - ⇒ TrainingBeta = n
  - Lookahead = [0.1s , 1s, 3s, 5s]
    - Time interval between the start and end of prediction
    - If we predict $\beta_{t+1} \sim \beta_{t+L}$ based on $\beta_{t-n+1} \sim \beta_t$
    - ⇒ Lookahead = L
  - Experiment with **6 models**
    - None (hold current β)
    - LR(Regularized Linear Regression)
    - SVR(Support Vector Regression)
    - Gpse(Gaussian Process with SE kernel)
    - GPrq(Gaussian Process with RQ kernel)
    - Gpsenoise(Gaussian Process with SE noise Kernel)
    - **Calculate Beta Difference for each model**
- **Predict Position $\boldsymbol{X_{t+1} \sim X_{t+L}}$**
  - Predicted Beta
  - Dynamics Equations
  - **Calculate Position Difference for each model**
- **Repeat experiments 10 times individually**

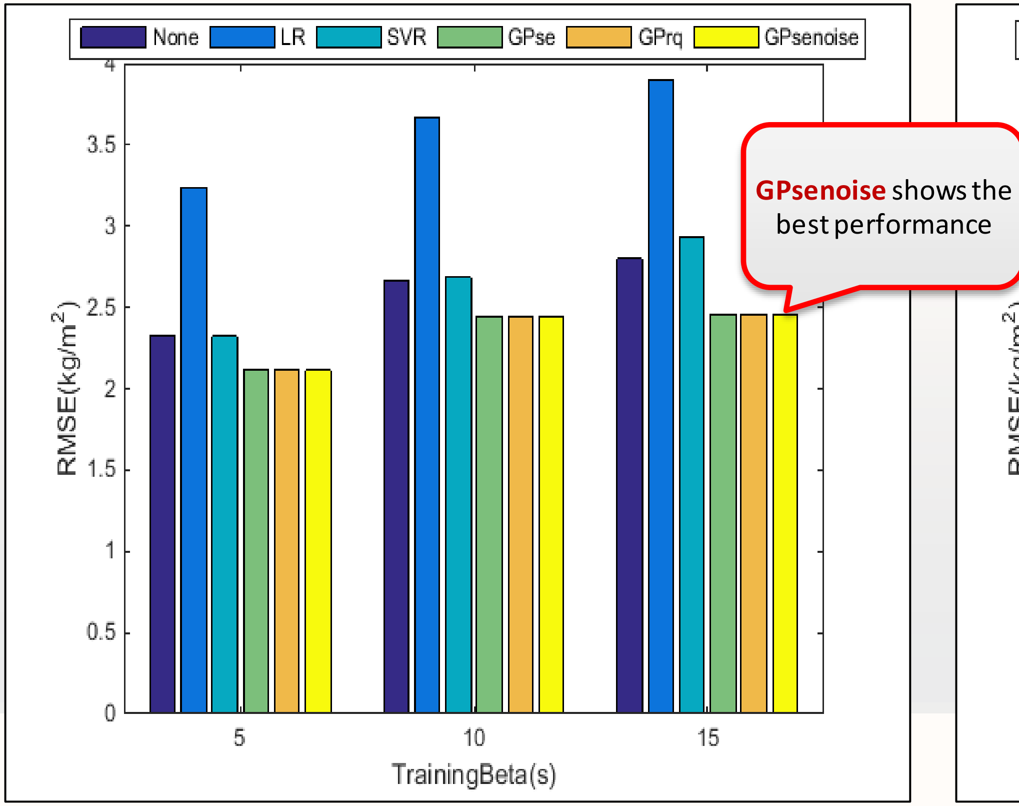
RESULTS – BETA

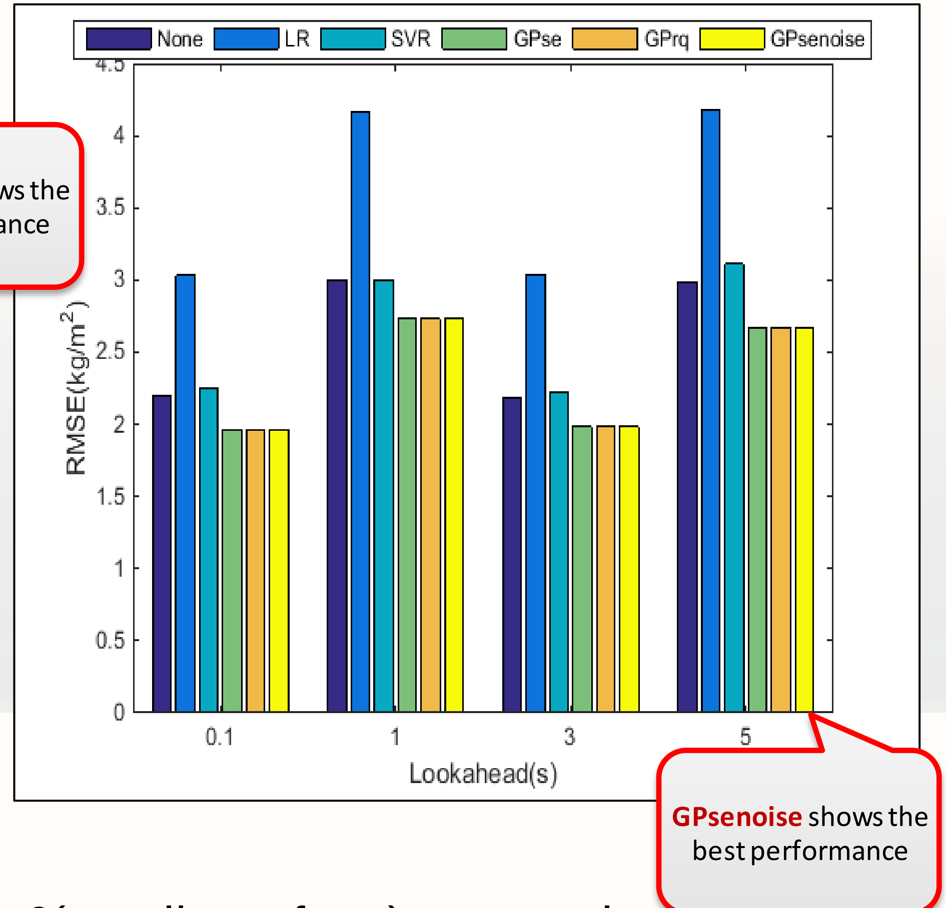


GP shows good result in 617sec~620sec (right after inflection point) remarkably

TrainingBeta : The number of recent beta used to training regression model
Lookahead : Time interval between the current and end of prediction

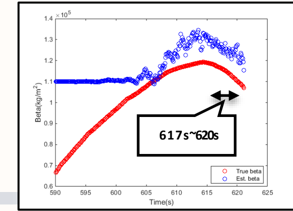
RESULTS – BETA





- Regression model based on GP predict β(non-linear form) accurately
- Gpsenoise performs well in all cases (every TrainingBeta , Lookahead settings)

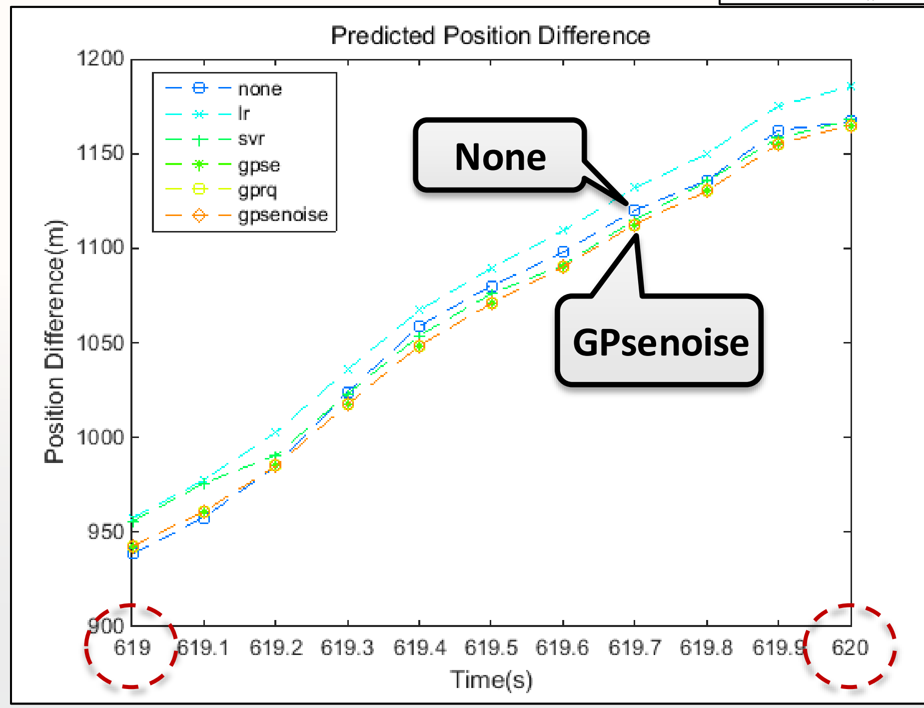
RESULTS – POSITION



	MAE	RMSE
None	341.52	385.3968
LR	343.4452	388.2391
SVR	341.4957	385.0937
GPse	341.0656	384.7834
GPrq	341.0688	384.7886
GPsenoise	**341.0656**	**384.7834**

- MAE = Mean Absolute Error
  - $\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$
- RMSE = Root Mean Square Error
  - $\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$

y_i : True value / $\hat{y}_i$: Predicted value



- GP regression predict relatively accurate position
  - **GPsenoise**>GPse>GPrq>SVR>None>LR

- GPsenoise shows the best performance in position difference
- But the improvement is not large (MAE : 0.5m , RMSE : 0.6m)

- It will be more useful
  - When the beta changes suddenly
  - For tracking a target which is strongly influenced by β

CONCLUSION

Experimental Setting

1) Beta / Position Estimation : IMM with UKF
2) Beta Prediction : Regression (Gaussian Process / Support Vector Regression / Regularized Linear Regression / None(Constant Prediction))
3) Position Prediction : Predicted Beta + Dynamics Equations

Beta prediction

- GP based regression model predict relatively exact β
  - Gpsenoise shows the best performance (SE kernel + Noise kernel)
  - β : Unknown Paramter to calculate aerodynamic drag acceleration
    - $a_D = -\frac{\rho}{2\beta}||v||v$

Position prediction

- GP based regression model predict relatively exact position
  - Gpsenoise shows the best performance (SE kernel + Noise kernel)
  - It is possible to predict position well by predicting β accurately

REFERENCE

[1] Farrell, William J. "Interacting multiple model filter for tactical ballistic missile tracking." Aerospace and Electronic Systems, IEEE Transactions on 44.2 (2008): 418-426.

[2] Song, Taek Lyul, Hyoung Won Kim, and Darko Mušicki. "Iterative joint integrated probabilistic data association." Information Fusion (FUSION), 2013 16th International Conference on. IEEE, 2013.

[3] Liu, Qiang, Xin Wang, and Nageswara SV Rao. "Artificial neural networks for estimation and fusion in long-haul sensor networks." Information Fusion (Fusion), 2015 18th International Conference on. IEEE, 2015.

[4] Farina, A., L. Timmoneri, and D. Vigilante. "Classification and launch-impact point prediction of ballistic target via multiple model maximum likelihood estimator (MM-MLE)." *Radar, 2006 IEEE Conference on*. IEEE, 2006.

[5] Jung, Jae-Kyung, and Dong-Hwan Hwang. "The novel impact point prediction of a ballistic target with interacting multiple models." *Control, Automation and Systems (ICCAS), 2013 13th International Conference on*. IEEE, 2013.

[6] Aly, Seham Mouawad, Raafat El Fouly, and Hoda Braka. "Extended Kalman filtering and Interacting Multiple Model for tracking maneuvering targets in sensor netwotrks." *Intelligent solutions in Embedded Systems, 2009 Seventh Workshop on*. IEEE, 2009.

[7] J. O. Ogutu, T. Schulz-Streeck, and H.-P. Piepho, “Genomic selection using regularized linear regression models: ridge regression, lasso, elastic net and their extensions,” in BMC proceedings, vol. 6, no. Suppl2. BioMed Central Ltd, 2012, p. S10.

[8] Bishop, Christopher M. "Pattern recognition." Machine Learning 128 (2006).

[9] D. Basak, S. Pal, and D. C. Patranabis, “Support vector regression,” Neural Information Processing-Letters and Reviews, vol. 11, no. 10, pp. 203–224, 2007.

[10] Murphy, Kevin P. Machine learning: a probabilistic perspective. MIT press, 2012.

[11] Advances in Gaussian Processes Tutorial at NIPS 2006 in Vancouver

[12] Ebden, M. "Gaussian Processes for Regression: A Quick Introduction.[online] Available at:< http://www. robots. ox. ac. uk/~ mebden/reports." GPtutorial. pdf (2008).

[13] Neil D. Lawrence and Raquel Urtasun , “Session 1: Gaussian Processes” , CVPR 2012 tutorial (http://www.cs.toronto.edu/~urtasun/tutorials/gp_cvpr12_session1.pdf)

[14] C. E. Rasmussen, “Gaussian processes for machine learning,” 2006.