Improved Speed Estimation of BLDC Motors using Gaussian Processes

Bachelor Student: Mariana Petrova

Supervisors: Ajit Basarur, Jana Mayer

Referee: Uwe D. Hanebeck

Intelligent Sensor-Actuator-Systems Laboratory (ISAS),
Institute for Anthropomatics and Robotics,
Karlsruhe Institute of Technology (KIT),
Karlsruhe, Germany

Karlsruhe Institute of Technology

http://isas.iar.kit.edu

Intelligent Sensor-Actuator-Systems

Introduction

Find the precise angular position and rotational speed of the rotor in order to achieve more accurate motor control.

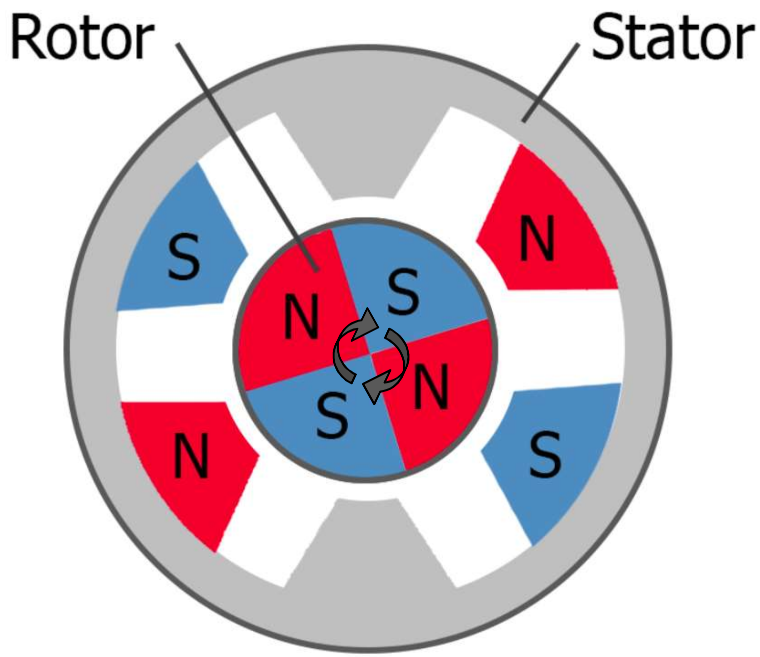
Permanent magnets producing magnetic field

↓

Estimate position and speed

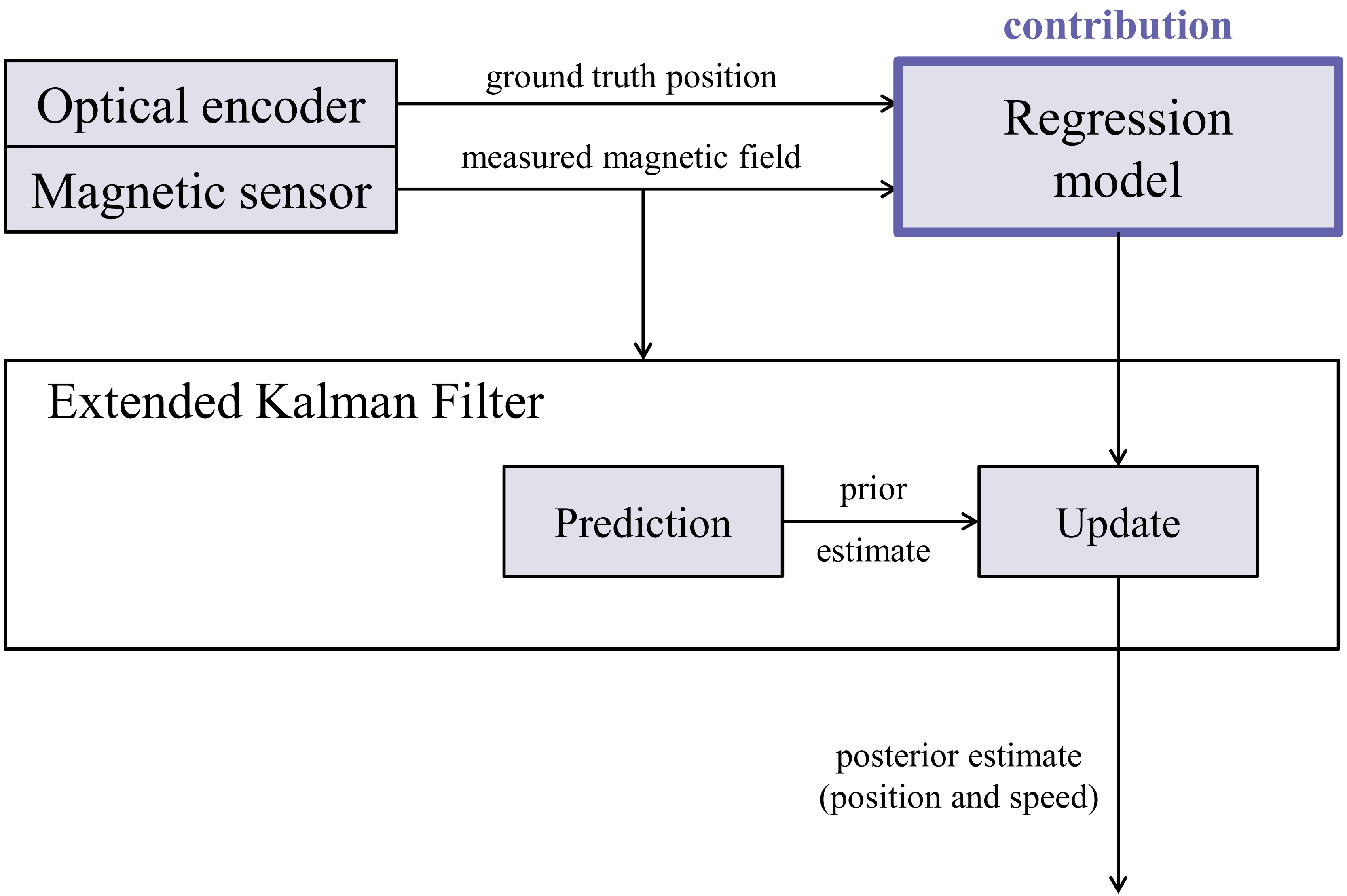
↓

Feedback for the electrical motor



Contents

Estimation Model

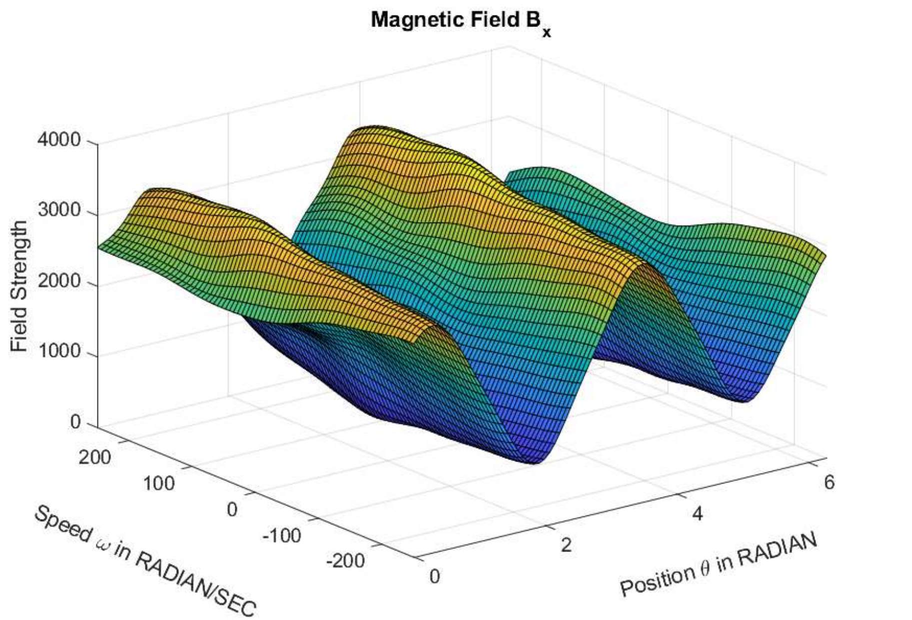


Gaussian Processes

Goal: Estimate a real-valued function f from a given data with inputs $\boldsymbol{X} = \{\underline{x}_1, \underline{x}_2, \dots, \underline{x}_N\}$ and observed outcomes $\underline{y}$

$$\underline{y} = f(\mathbf{X}) + \epsilon, \quad \text{with } \epsilon \sim N(0, \sigma_n^2) \text{ and}$$

$$f : \underbrace{\mathcal{X}}_{(angle,\,speed)} \rightarrow \underbrace{\mathcal{Y}}_{(magnetic\ field)}$$



A **Gaussian process** defines a distribution over functions - $p(f)$

Define $\underline{x}_*$ for which we would like to estimate $f(x_*)$

So we are trying to get the predictive posterior distribution

$$p(y_*|\underline{x}_*, \boldsymbol{X}, \underline{y}) = N(\boldsymbol{\mu}_*, \boldsymbol{\sigma}_*^2)$$

And we assume that y and y_* together are jointly Gaussian.

$$\boldsymbol{\mu}_* = \underline{k}(\mathbf{X}, \underline{x}_*)^T * (k(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I})^{-1} * \underline{y}$$

$$\boldsymbol{\sigma}_*^2 = k(\underline{x}_*, \underline{x}_*) - \underline{k}(\mathbf{X}, \underline{x}_*)^T * (k(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I})^{-1} * \underline{k}(\mathbf{X}, \underline{x}_*) + \sigma_n^2 \mathbf{I}$$

Gaussian Process - Kernels

Kernel: defines the similarity of two values of a function calculated at two different locations in the input space.

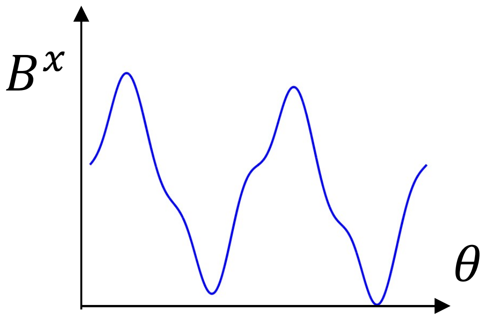
Controlled by hyper-parameters, collected in $\underline{\vartheta}$

The previously used covariance function $k(\underline{x}, \underline{x}_*)$ is :

$$k(\underline{x}, \underline{x}_*) = \sigma^2 * \exp\left\{-\frac{1}{2}\left[\frac{(\theta - \theta_*)^2 + (\omega - \omega_*)^2}{2l^2}\right]\right\}$$

$$\text{where } \underline{x} = \begin{pmatrix}\theta \\ \omega\end{pmatrix}, \quad \underline{x}_* = \begin{pmatrix}\theta_* \\ \omega_*\end{pmatrix}, \quad \underline{\vartheta} = \{\sigma^2, l\}$$

Previous Regression Model

Different input scales	Data re-scaling $\omega_{new} = \frac{\omega \ - \min(\omega)}{\max(\omega) - \min(\omega)} \cdot 2\pi$
Not consider the periodic structure 	Different kernel function
Poor scaling with training data in $O(N^3)$	Parametric regression approaches

Improved Kernels

Previously: one length scale for both θ and ω.

Now: Achieve better flexibility by multiplying kernels defined on each individual input.

SE-ARD :

$$k(\underline{x}, \underline{x}_*) = \sigma^2 * \exp\left\{-\frac{1}{2}\left[\frac{(\theta - \theta_*)^2}{2l_\theta^2} + \frac{(\omega - \omega_*)^2}{2l_\omega^2}\right]\right\}$$

Improved Kernels

Previously: one length scale for both θ and ω.

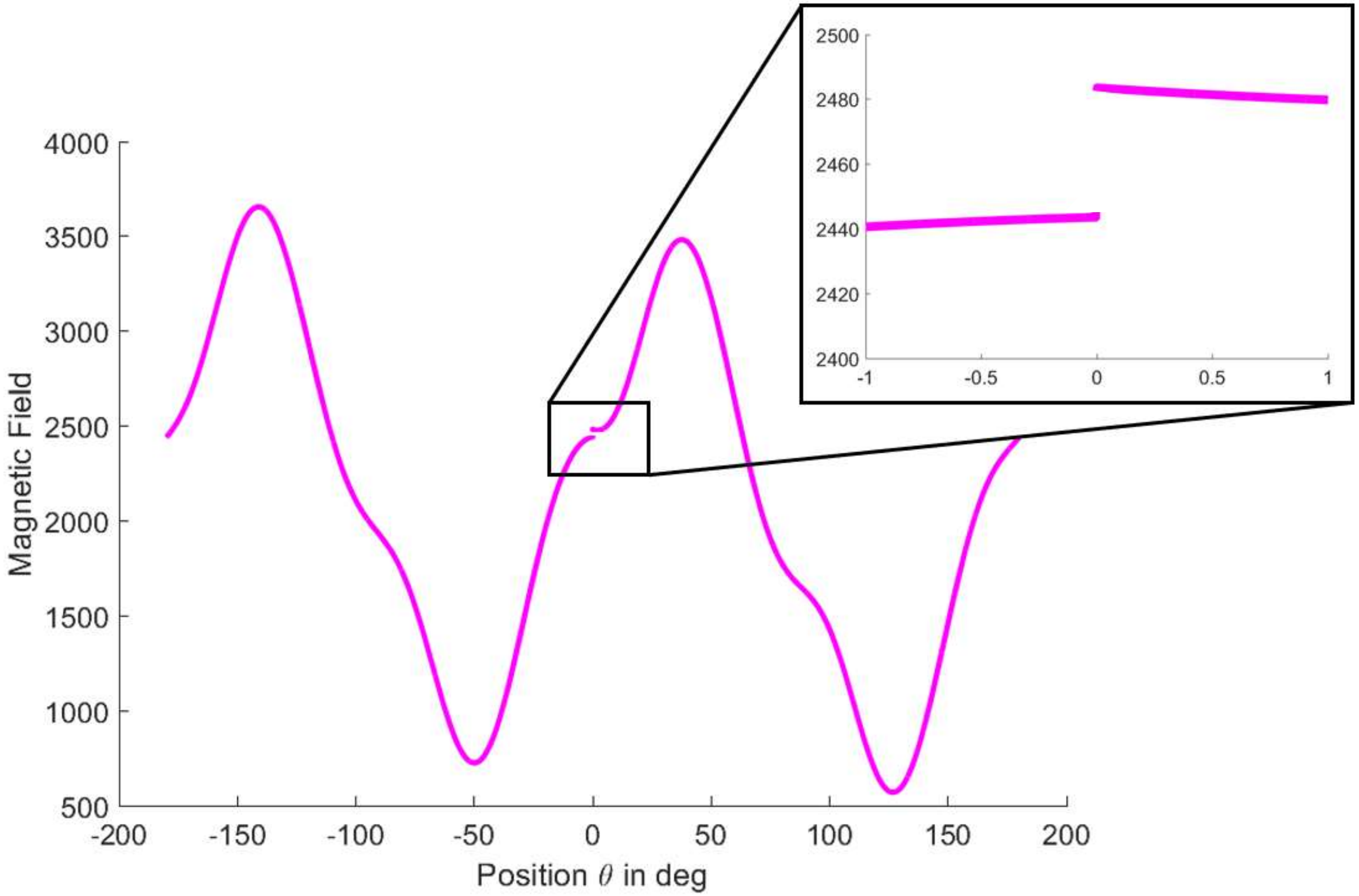
Now: Achieve better flexibility by multiplying kernels defined on each individual input.

SE-ARD :

$$k(\underline{x}, \underline{x}_*) = \sigma^2 * \exp\left\{-\frac{1}{2}\left[\frac{(\theta - \theta_*)^2}{\boxed{2l_\theta^2}} + \frac{(\omega - \omega_*)^2}{\boxed{2l_\omega^2}}\right]\right\}$$

Improved kernels

SE-ARD $$k(\underline{x}, \underline{x}_*) = \sigma^2 * \exp\left\{-\frac{1}{2}\left[\frac{(\theta-\theta_*)^2}{2l_\theta^2} + \frac{(\omega-\omega_*)^2}{2l_\omega^2}\right]\right\}$$



Improved Kernels

What about the periodicity of the magnetic field with θ?

PER-ARD :

$$k(\underline{x}, \underline{x}_*) = \sigma^2 \exp\left\{-\frac{2\sin\left(\frac{|\theta - \theta_*|}{2}\right)}{l_\theta^2}\right\} \exp\left\{-\frac{(\omega - \omega_*)^2}{2 l_\omega^2}\right\}$$

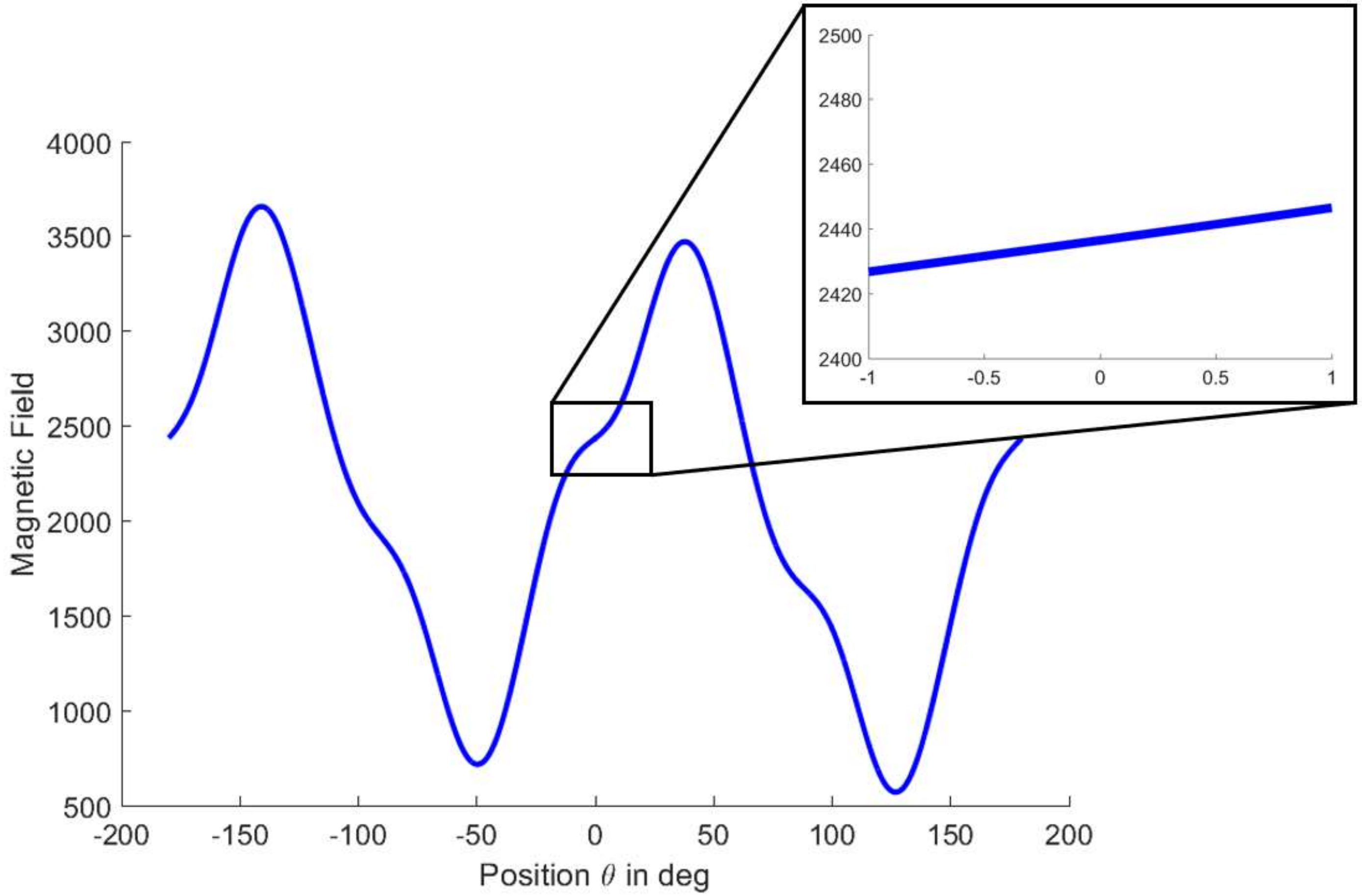
What about the periodicity of the magnetic field with θ?

PER-ARD :

$$k(\underline{x}, \underline{x}_*) = \sigma^2 \exp\left\{-\boxed{\frac{2\sin\left(\frac{|\theta - \theta_*|}{2}\right)}{l_\theta^2}}\right\} \exp\left\{-\frac{(\omega - \omega_*)^2}{2l_\omega^2}\right\}$$

Improved kernels

PER-ARD $$k(\underline{x}, \underline{x}_*) = \sigma^2 \exp\left\{-\frac{2\,\mathrm{si}\left(\frac{|\theta-\theta_*|}{2}\right)}{l_\theta^2}\right\} + \sigma^2 \exp\left\{-\frac{(\omega-\omega_*)^2}{2 l_\omega^2}\right\}$$



More Scalable Regression Approaches

Non-parametric methods (like Gaussian Processes) scale with the number of training points $\left(O(N^3)\right)$

$\Rightarrow$ are not very data efficient.

This is a restriction for our model, since great amount of data can be generated by the hardware setup.

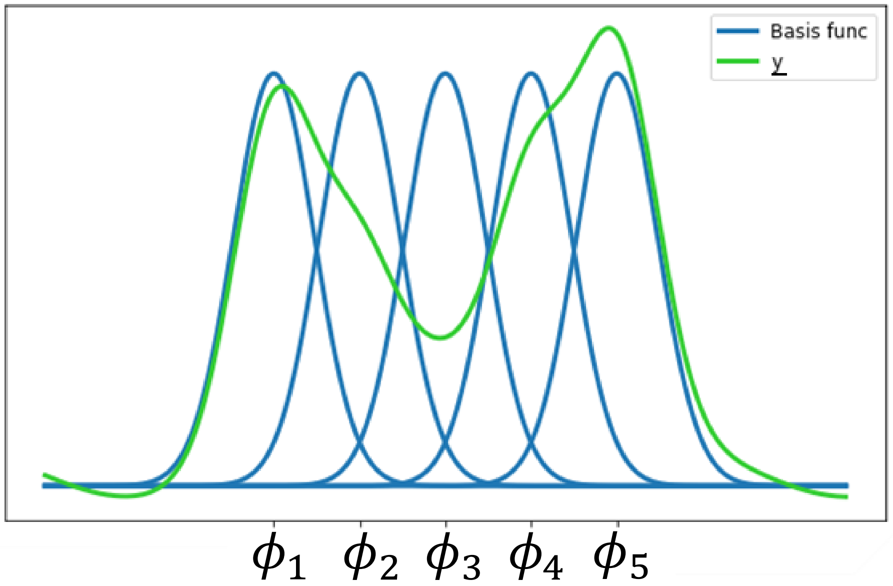
$\Rightarrow$ Solution: parametric methods.

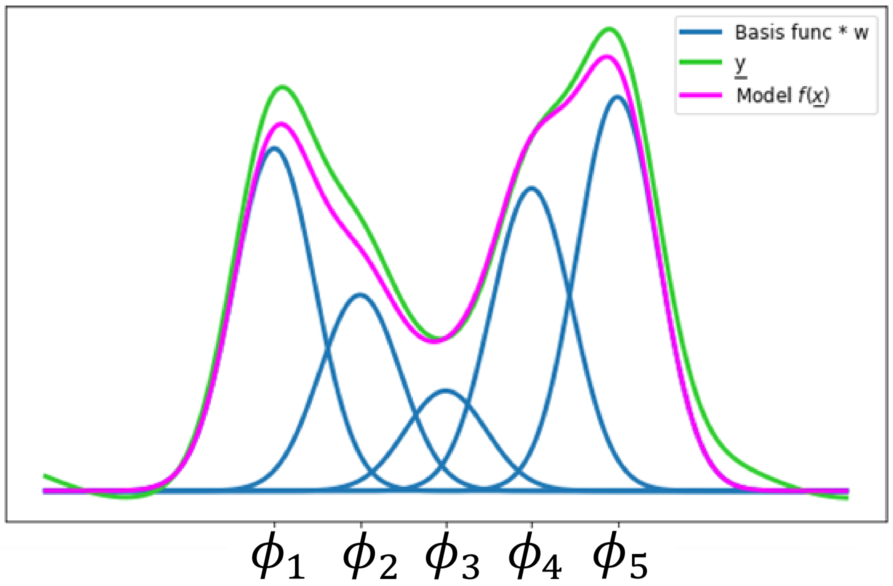
Linear Regression with Basis Functions

Consider M basis points $\mathbf{X}_b = (\underline{x}_{b1}, \underline{x}_{b2}, \dots, \underline{x}_{bM})$

We define a linear model with a weight vector $\underline{w} \in \mathbb{R}^M$:

$$f_w(\underline{x}) = \underline{\varphi}(\underline{x})^T \underline{w} \quad \Rightarrow \quad f_w(\underline{x}) = \underline{k}(\mathbf{X}_b, \underline{x})^T \underline{w}$$





Use the set of training points $\mathbf{X} = \left(\underline{x}_1, \underline{x}_2, \dots, \underline{x}_M\right)$ with the corresponding observed outputs $\underline{y}$

$$\mathbf{\Phi} = \underbrace{\begin{bmatrix} \underline{k}\left(\mathbf{X}_b, \underline{x}_1\right)^T \\ \vdots \\ \underline{k}\left(\mathbf{X}_b, \underline{x}_N\right)^T \end{bmatrix}}_{NxM\ matrix}$$

$$\underline{y} = f_w(\mathbf{X}) + \epsilon \quad \text{with} \quad f_w(\mathbf{X}) = \mathbf{\Phi}\,\underline{w}$$

How do we calculate $\underline{w}$?

$$
\begin{aligned}
\underline{w}_{MAP} &= \underset{\underline{w}}{\operatorname{argmax}}\, p\left(\underline{w} \,\middle|\, \mathbf{X}, \underline{y}\right) \\
&\propto p\left(\underline{y} \mid \mathbf{X}, \underline{w}\right) \cdot p(\underline{w}) \\
&\propto \underset{\underline{w}}{\operatorname{argmax}}\, \log p\left(\underline{y} \mid \mathbf{X}, \underline{w}\right) \cdot \log p(\underline{w}) \\
&= \underset{\underline{w}}{\operatorname{argmin}}\, \frac{1}{2\sigma^2} \|\underline{y} - \mathbf{\Phi}\underline{w}\|_2^2 + \frac{1}{2\sigma_w^2} \|\underline{w}\|_2^2
\end{aligned}
$$

How do we calculate $\underline{w}$?

$$\underline{w}_{MAP} = \underset{\underline{w}}{\operatorname{argmax}}\, p\left(\underline{w} \,\middle|\, \mathbf{X}, \underline{y}\right)$$

$$\propto p\left(\underline{y} \mid \mathbf{X}, \underline{w}\right) \cdot p(\underline{w})$$

$$\propto \underset{\underline{w}}{\operatorname{argmax}} \log p\left(\underline{y} \mid \mathbf{X}, \underline{w}\right) \cdot \log p(\underline{w})$$

$$= \underset{\underline{w}}{\operatorname{argmin}} \frac{1}{2\sigma^2} \|\underline{y} - \boldsymbol{\Phi}\underline{w}\|_2^2 + \frac{1}{2\sigma_w^2} \|\underline{w}\|_2^2$$

$$\boxed{\underline{w}_{MAP} = (\boldsymbol{\Phi}^{\mathrm{T}}\, \boldsymbol{\Phi} + \lambda \cdot \mathrm{I})^{-1} \boldsymbol{\Phi}^{\mathbf{T}} \underline{\mathrm{y}}}$$

Sparse Gaussian Processes using Pseudo-inputs

The model in Sparse Gaussian Processes using Pseudo-inputs [1] (SPGP) takes $M \ll N$ pseudo-points, where N is the number of real data points.

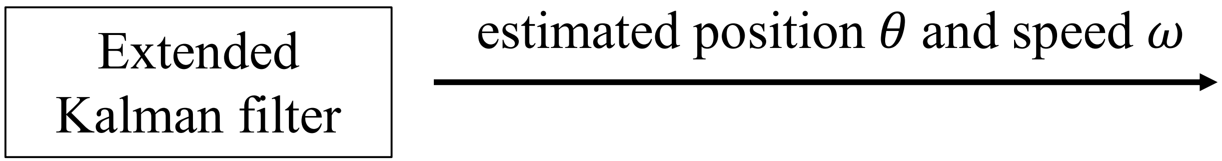
The pseudo-points locations are optimized together with the hyper-parameters in the kernel.

Gaussian Processes	SPGP
$O(N^3)$	$O(MN^2)$

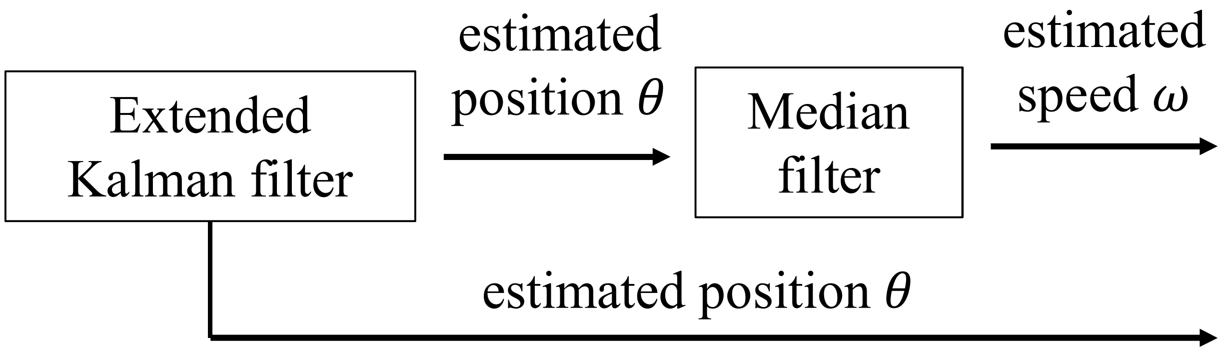
[1] Snelson, E. and Ghahramani, Z. (2006). Sparse Gaussian processes using pseudo-inputs

An Alternative Speed Estimation

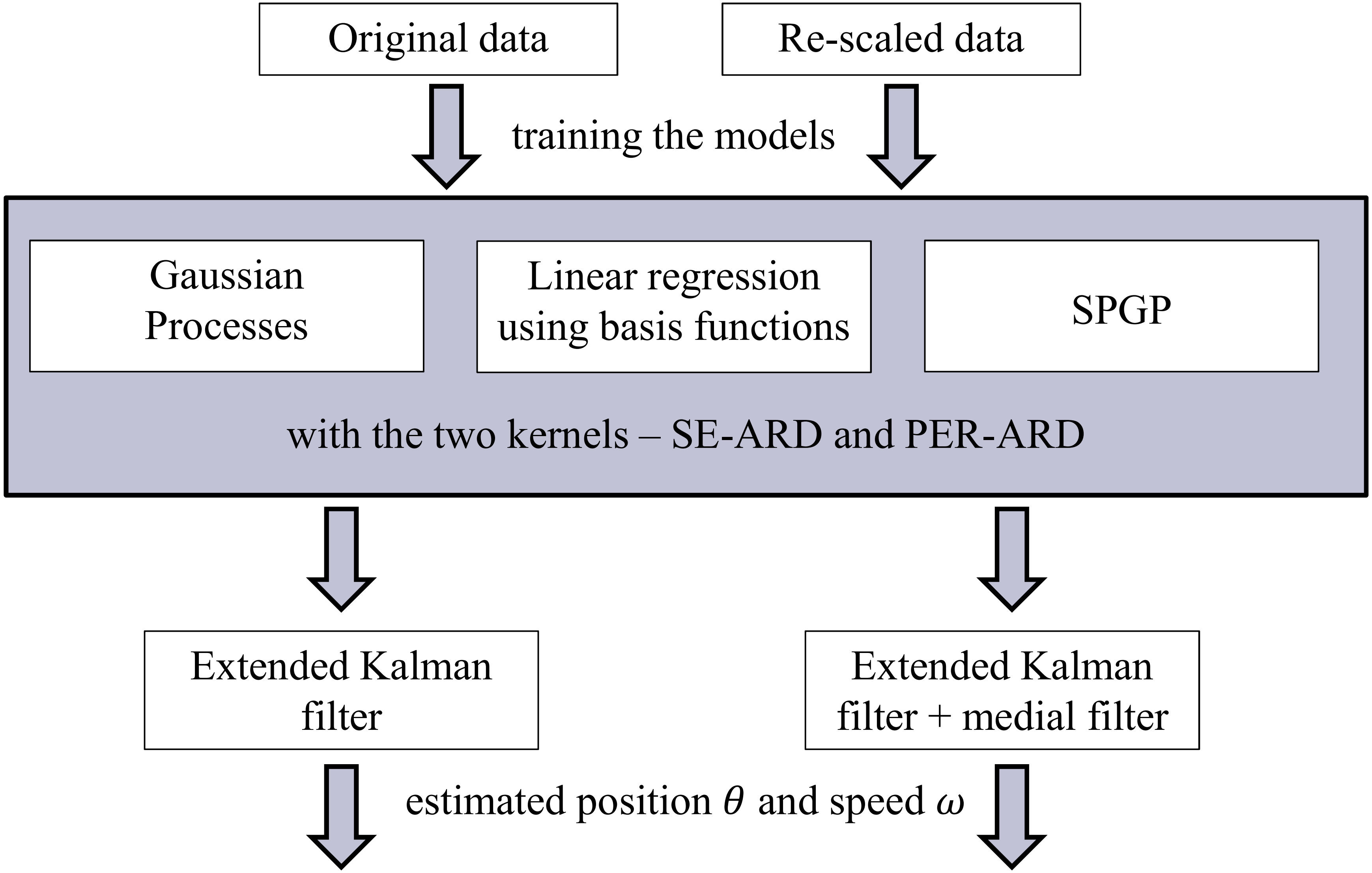
First estimation approach



Second estimation approach



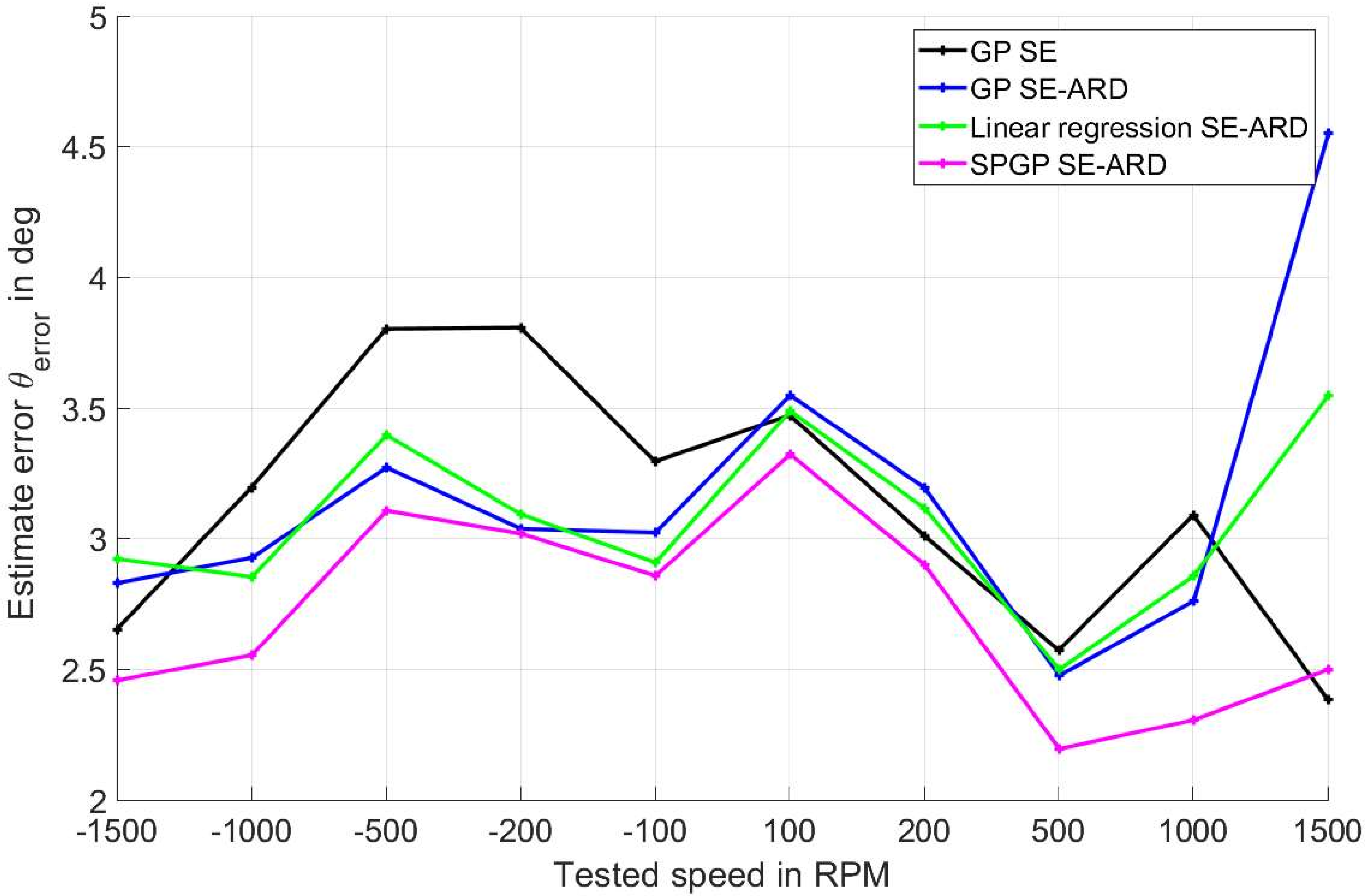
All models

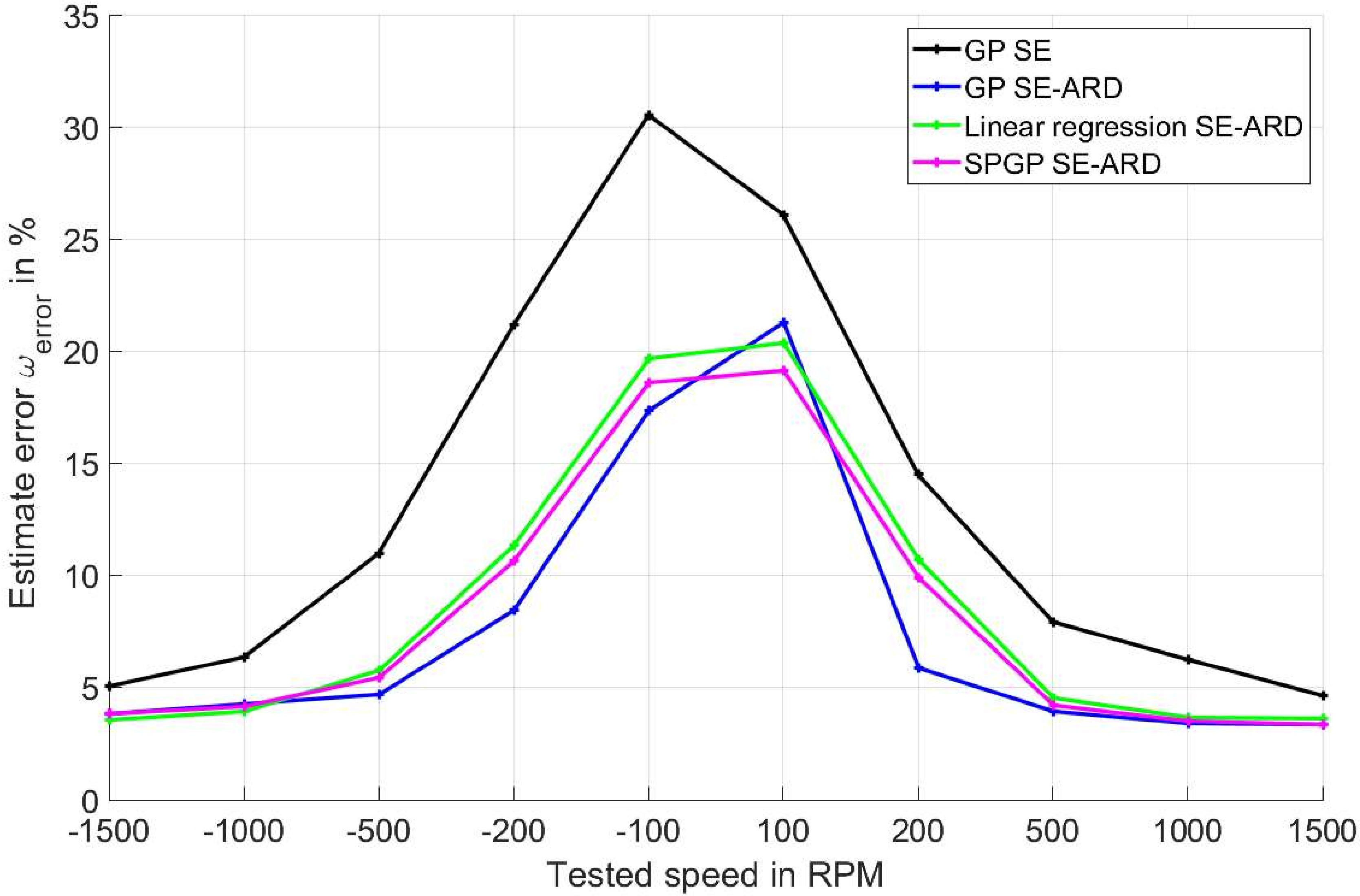

Original data
Re-scaled data
training the models
Gaussian Processes
Linear regression using basis functions
SPGP
with the two kernels – SE-ARD and PER-ARD
Extended Kalman filter
Extended Kalman filter + medial filter
estimated position θ and speed ω

Evaluation

Tested speed / RPM	RMSE on	GP SE (original data, EKF)	GP SE-ARD (original data, EKF)	Linear regr. SE-ARD (original data, EKF)	SPGP SE-ARD (original data, EKF)
all speeds	θ / deg	3.1406	3.3207	3.1219	2.773
	ω/RPM	51.2665	37.1094	36.2002	36.2694

		GP PER-ARD (original data, EKF)	GP SE-ARD (rescaled data, EKF + median filter)	Linear regr. PER-ARD (original data, EKF)	SPGP SE-ARD (rescaled data, EKF + median filter)
all speeds	θ / deg	7.6937	3.0862	3.0831	2.7836
	ω/RPM	39.5337	50.8665	52.5253	42.6009


GP SE
GP SE-ARD
Linear regression SE-ARD
SPGP SE-ARD
Estimate error θ_{error} in deg
Tested speed in RPM
5
4.5
4
3.5
3
2.5
2
-1500
-1000
-500
-200
-100
100
200
500
1000
1500


GP SE
GP SE-ARD
Linear regression SE-ARD
SPGP SE-ARD
Estimate error ω_{error} in %
Tested speed in RPM
0
5
10
15
20
25
30
35
-1500
-1000
-500
-200
-100
100
200
500
1000
1500

Conclusion

- Address the main flaws of previous works.
- An emphasis was put on better utilization of the training data.

Result: Identified several models showing a better state estimation performance than previous works.

Future Work

1. Substituting the radial basis functions with trigonometric ones.

2. Substitute the Kalman Filter in favour of a machine learning approach that considers previous state or context.
   - i.e. Long Short-Term Memory (LSTM)

3. Active learning framework (uncertainty sampling).

Thank you for your attention

Intelligent

Sensor-Actuator-Systems

Appendix

Linear Regression in Detail

$$\begin{aligned}
\underline{\hat{w}} &= \arg\max_{\underline{w}} \quad p(\underline{w} \mid \mathcal{D}) \\
&= \arg\max_{\underline{w}} \quad \frac{p(\mathcal{D} \mid \underline{w}) \cdot p(\underline{w})}{p(\mathcal{D})} \\
&= \arg\max_{\underline{w}} \quad p(\mathcal{D} \mid \underline{w}) \cdot p(\underline{w}) \\
&= \arg\max_{\underline{w}} \quad \log p(\underline{y} \mid \mathbf{X}, \underline{w}) + \log p(\underline{w}) \\
&= \arg\min_{\underline{w}} \quad \frac{1}{2}(\underline{y} - \mathbf{\Phi}\underline{w})^T \Sigma_y^{-1} (\underline{y} - \mathbf{\Phi}\underline{w}) + \frac{1}{2} w^T \Sigma_w^{-1} w.
\end{aligned}$$

$$\underline{\hat{w}} = \arg\min_{\underline{w}} \quad \frac{1}{2\sigma_y^2} \|\underline{y} - \mathbf{\Phi}\underline{w}\|_2^2 + \frac{1}{2\sigma_w^2} \|\underline{w}\|_2^2.$$

$$\underline{y} \sim \mathcal{N}\left(f_w(\mathbf{X}), \sigma_y^2 \cdot \mathbf{I}\right) \qquad \sigma_y^2 \cdot \mathbf{I} = \Sigma_y$$

$$\underline{w} \sim \mathcal{N}(0, \sigma_w^2 \cdot \mathbf{I}) \qquad \sigma_w^2 \cdot \mathbf{I} = \Sigma_w.$$

Linear Regression in Detail

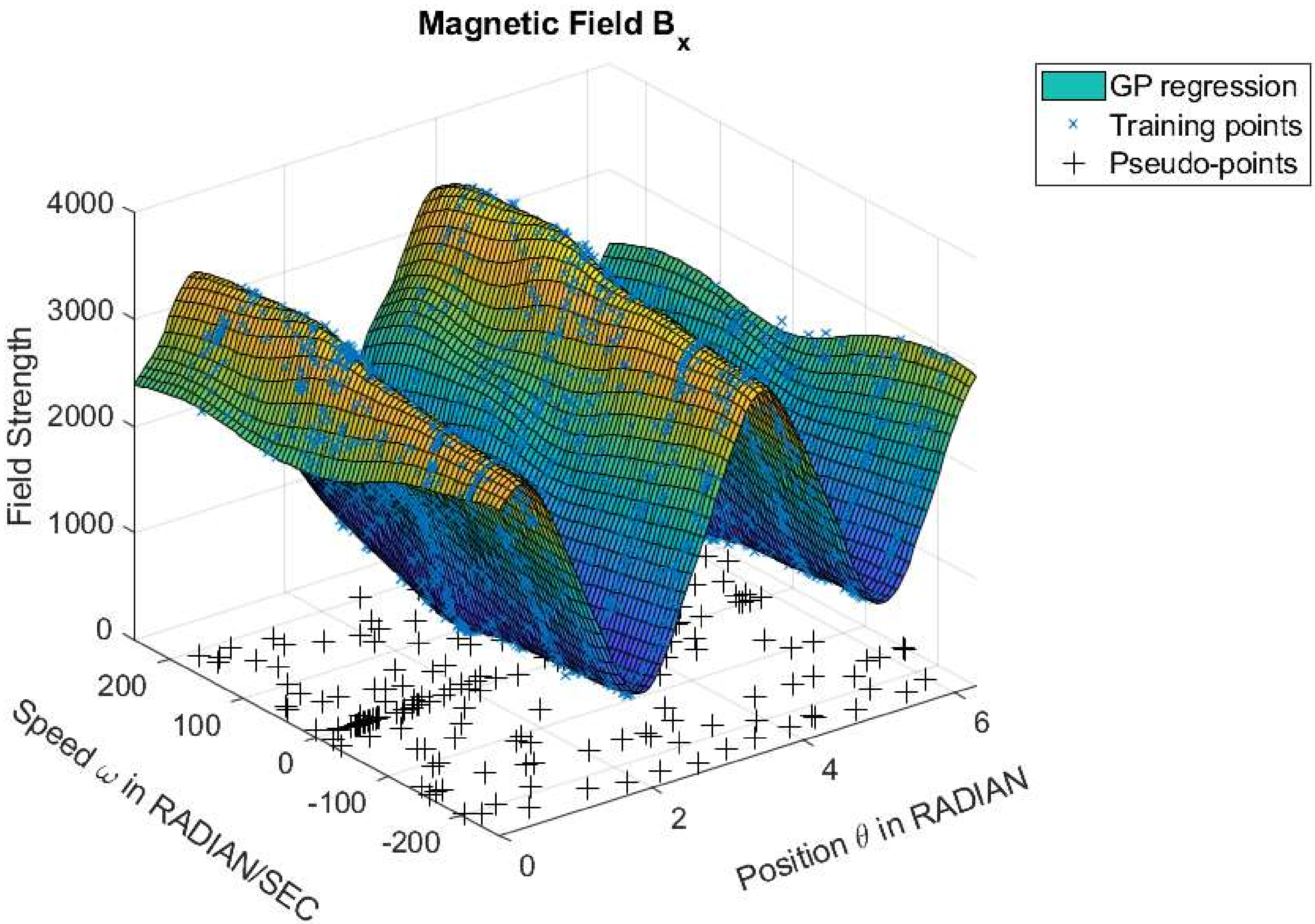
$\frac{1}{\sigma_w^2}$ and $\frac{1}{\sigma_y^2}$ are only weights and are summarized in λ

The solution for the MAP estimate, can be derived analytically in closed form by setting the gradient with respect to $\underline{w}$ to $\underline{0}$.

$$\begin{aligned}
\underline{0} &= \nabla_{\underline{w}} \left(\frac{1}{2} \left\| \underline{y} - \mathbf{\Phi} \underline{w} \right\|_2^2 + \frac{\lambda}{2} \left\| \underline{w} \right\|_2^2 \right) \\
&= \mathbf{\Phi}^T \left(\underline{y} - \mathbf{\Phi} \underline{w} \right) + \lambda \cdot \underline{w} \\
&= \mathbf{\Phi}^T \underline{y} - \mathbf{\Phi}^T \mathbf{\Phi} \underline{w} + \lambda \cdot \underline{w} \\
&= \mathbf{\Phi}^T \underline{y} - (\mathbf{\Phi}^T \mathbf{\Phi} - \lambda \cdot \mathbf{I}) \underline{w}
\end{aligned}$$

$$\boxed{\hat{\underline{w}} = (\mathbf{\Phi}^T \mathbf{\Phi} + \lambda \cdot \mathbf{I})^{-1} \mathbf{\Phi}^T \underline{y}.}$$

SPGP



Magnetic Field B_x
GP regression
Training points
Pseudo-points
4000
3000
2000
1000
0
Field Strength
200
100
0
-100
-200
Speed ω in RADIAN/SEC
0
2
4
6
Position θ in RADIAN

Evaluation of the Regression Models

GP, SE-ARD			
	Tested speed ω_{test}/RPM	RMSE(B^x)	RMSE(B^y)
original data	all speeds	24.0851	29.4115
	100	31.3455	29.5093
	-100	31.9589	34.0501
	200	31.2527	12.6483
	-200	37.043	36.1389
	500	16.1633	20.4691
	-500	19.1951	21.2171
	1000	30.3442	26.471
	-1000	18.9244	18.5268
	1500	25.6991	32.4093
	-1500	12.8694	13.2978
rescaled data	all speeds	22.738	23.3306
	100	30.7039	31.7805
	-100	27.3712	31.1648
	200	31.0714	13.5354
	-200	33.5892	33.3805
	500	16.9664	20.8776
	-500	28.4761	20.529
	1000	22.9061	21.9617
	-1000	19.0699	19.0411
	1500	15.5473	19.9976
	-1500	13.1341	12.9673

Evaluation of the Regression Models

Linear regression, SE-ARD			
	Tested speed ω_{test}/RPM	RMSE(B^x)	RMSE(B^y)
original data	all speeds	20.3572	20.3436
	100	26.6053	27.7138
	-100	26.6358	27.5146
	200	27.3728	11.4616
	-200	32.7014	30.6205
	500	14.573	16.4511
	-500	18.789	19.0015
	1000	20.4266	19.2807
	-1000	17.2019	19.9658
	1500	13.2834	15.0703
	-1500	11.8905	12.3322
rescaled data	all speeds	20.4093	21.537
	100	24.9558	23.6075
	-100	24.4855	23.8256
	200	26.1694	10.7595
	-200	31.4702	30.2521
	500	13.2536	16.3365
	-500	16.2811	17.3904
	1000	21.1124	22.4258
	-1000	16.6563	21.2493
	1500	14.8652	16.6138
	-1500	14.7033	16.3391

Evaluation

Tested speed / RPM	RMSE on	GP SE	GP SE-ARD	Linear regression SE-ARD	SPGP SE-ARD
all speeds	θ / deg	3.1406	3.3207	3.1219	2.773
	ω / RPM	51.2665	37.1094	36.2002	36.2694
100	θ / deg	3.4722	3.5463	3.4869	3.3208
	ω / RPM	26.084 ≈ 26%	21.28 ≈ 21%	20.389 ≈ 20%	19.127 ≈ 19%
200	θ / deg	3.012	3.1956	3.1138	2.9006
	ω / RPM	28.989 ≈ 14%	11.735 ≈ 6%	21.365 ≈ 11%	19.759 ≈ 10%
500	θ / deg	2.5728	2.477	2.5006	2.1953
	ω / RPM	39.667 ≈ 8%	19.841 ≈ 4%	22.598 ≈ 5%	21.052 ≈ 4%
1000	θ / deg	3.0904	2.7623	2.8582	2.306
	ω / RPM	62.543 ≈ 6%	33.978 ≈ 3%	36.839 ≈ 4%	35.273 ≈ 4%

Data Re-scaling

Gaussian processes regression is prone to numerical problems as we have to inverse ill-conditioned covariance matrix.

The two inputs (angle, speed) are on vastly different scales.

$\Longrightarrow$ bring the two input spaces in the same scale

$$\omega_{new} = \frac{\omega - \min(\omega)}{\max(\omega) - \min(\omega)} \cdot 2\pi$$

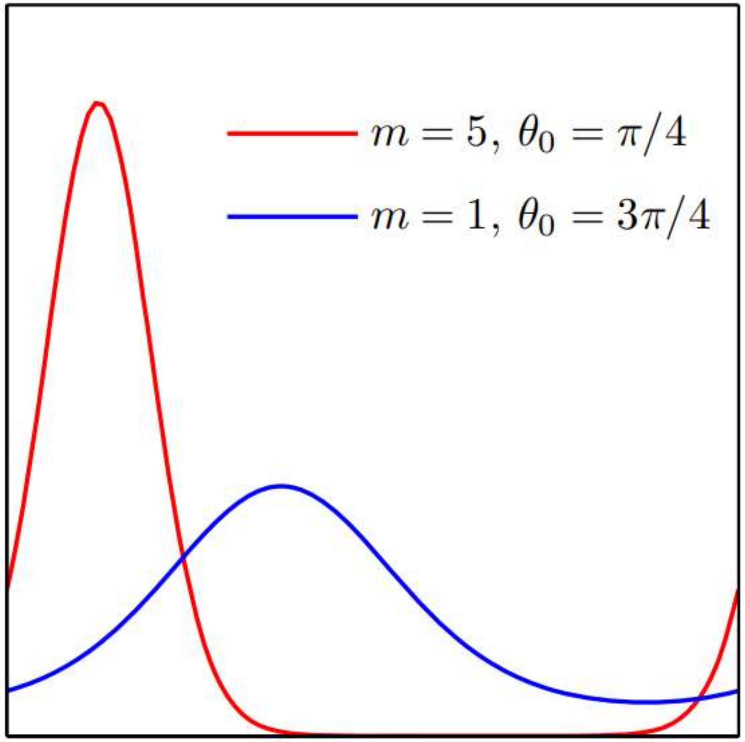
Additionally, zero mean output:

$$B^x = B^x - \text{mean}(B^x)$$
$$B^y = B^y - \text{mean}(B^y)$$

Von Mises Distribution

$$p(\theta|\theta_0, m) = \frac{1}{2\pi I_0(m)} \exp\{m \cos(\theta - \theta_0)\}$$

for large m, the distribution becomes approximately Gaussian.



Christopher M. Bishop - Pattern Recognition and Machine Learning