Gaussian Process Generative Models for Language and Robotics

Daichi Mochihashi
The Institute of Statistical Mathematics
daichi@ism.ac.jp

NAIST Data Science Lecture
2021-10-25 (Mon)
(based on Tutorial at CoRL 2019)

Self introduction

- PhD NAIST, 2005
- B.Sc. University of Tokyo, 1998
- Associate Professor at The Institute of Statistical Mathematics (ISM), Tokyo

- Research fields:
  Natural Language Processing and Bayesian Machine Learning

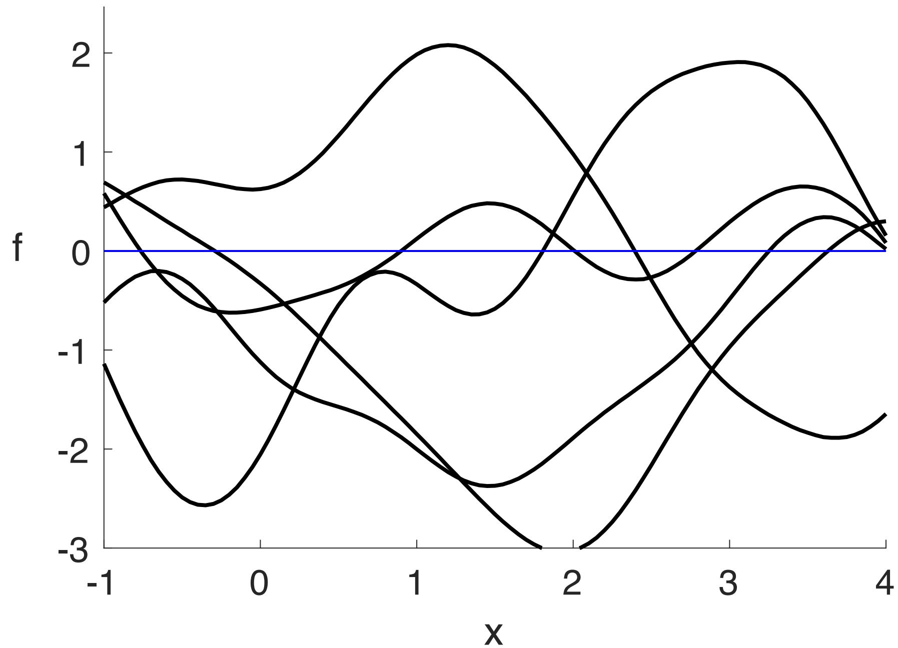
Overview

- Brief introduction to Gaussian processes (GP)
  - Very flexible nonlinear regression from Bayesian point of view
- Recognizing “Motion” (run, throw, bend, ...) through a combination of:
  - Gaussian processes + semi-Markov HMM + Hierarchical Dirichlet processes (IROS 2018)
  - Variational Antoencoder for high-dimensional signals (IROS 2019)
- GP Neural Networks for Text Visualization

Part 1:
What is a Gaussian process?

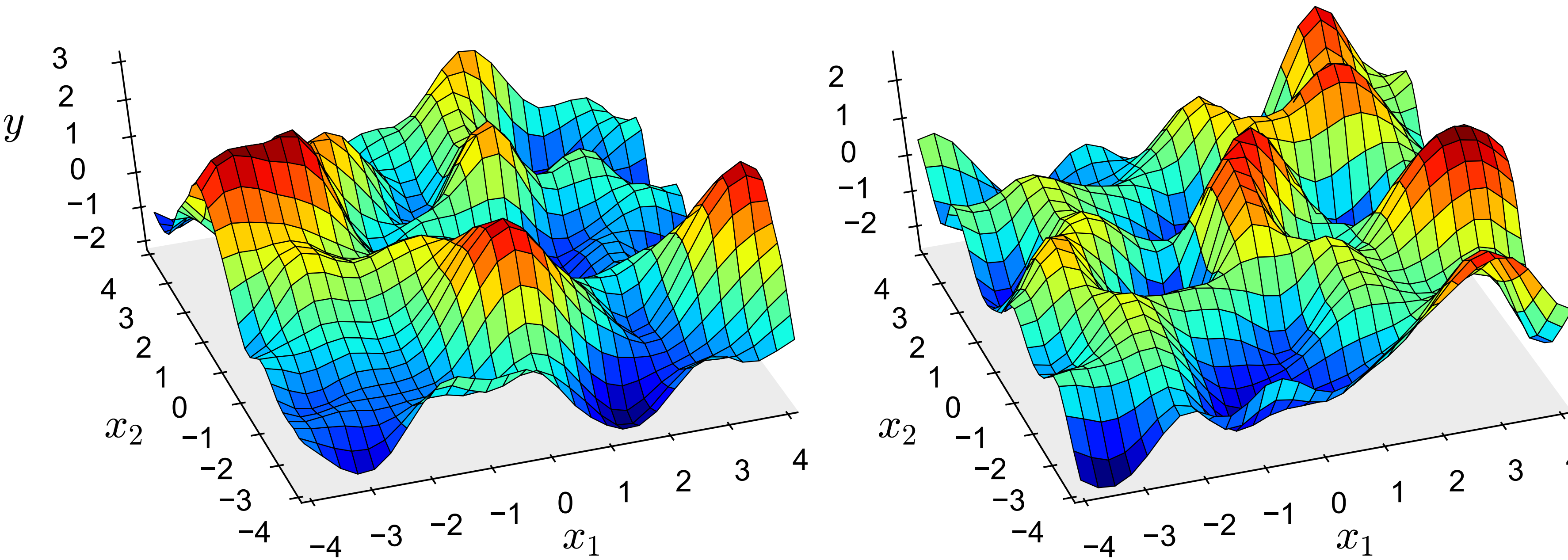
Gaussian processes

- Gaussian process···stochastic process to generate random functions



  – function : mapping from $\mathbf{x} \mapsto y$
  – As a function of time *t*, it represents a trajectory

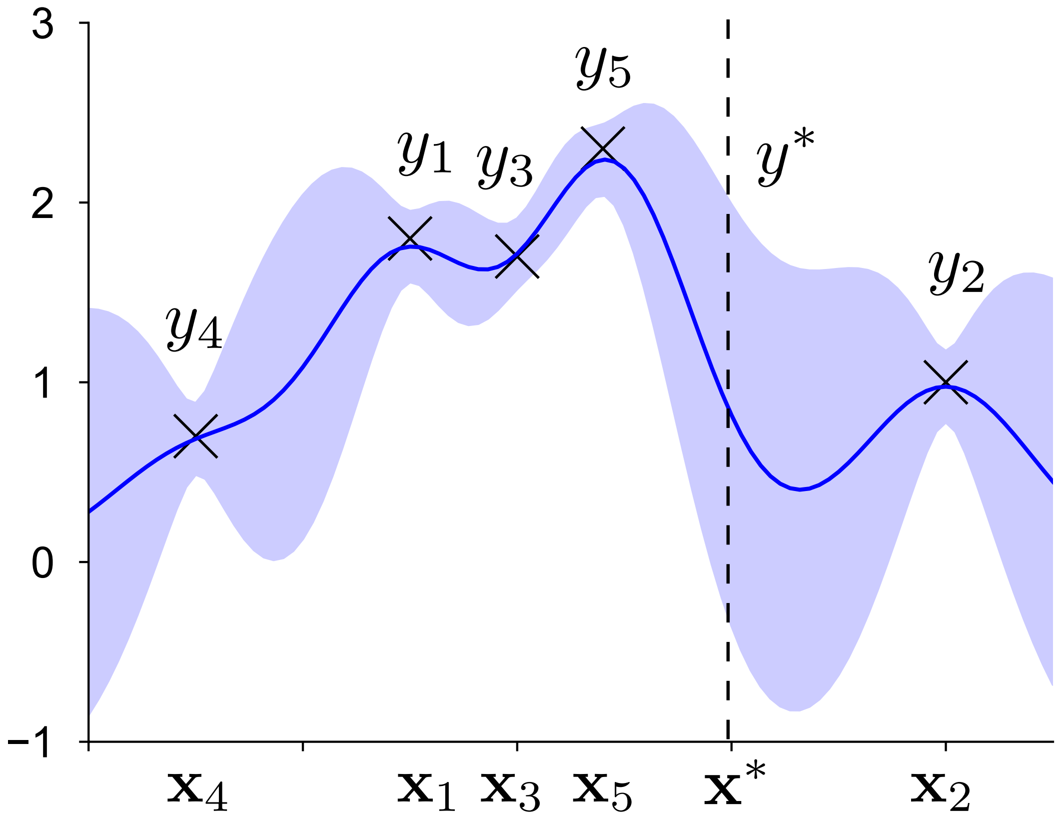
Gaussian processes (2)



- Gaussian process on two-dimensional inputs
  ＝Random surface function
- Similarly for higher-dimensional inputs *x*

Gaussian processes (3)

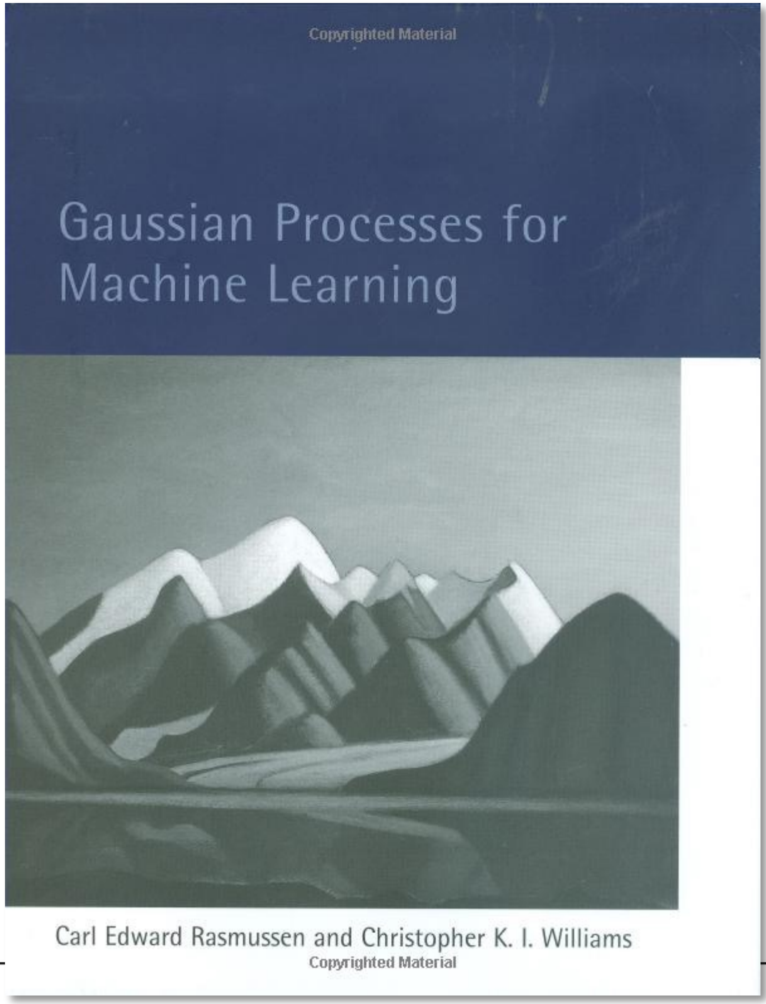
- Bayesian learning of a function: given datapoint, posterior distribution of functions is obtained



- Blue: Expectation
- High variance (light blue) for area with no data
- Ordinary model cannot represent variances of prediction

GPML

- GPML ("Gaussian Processes for Machine Learning")



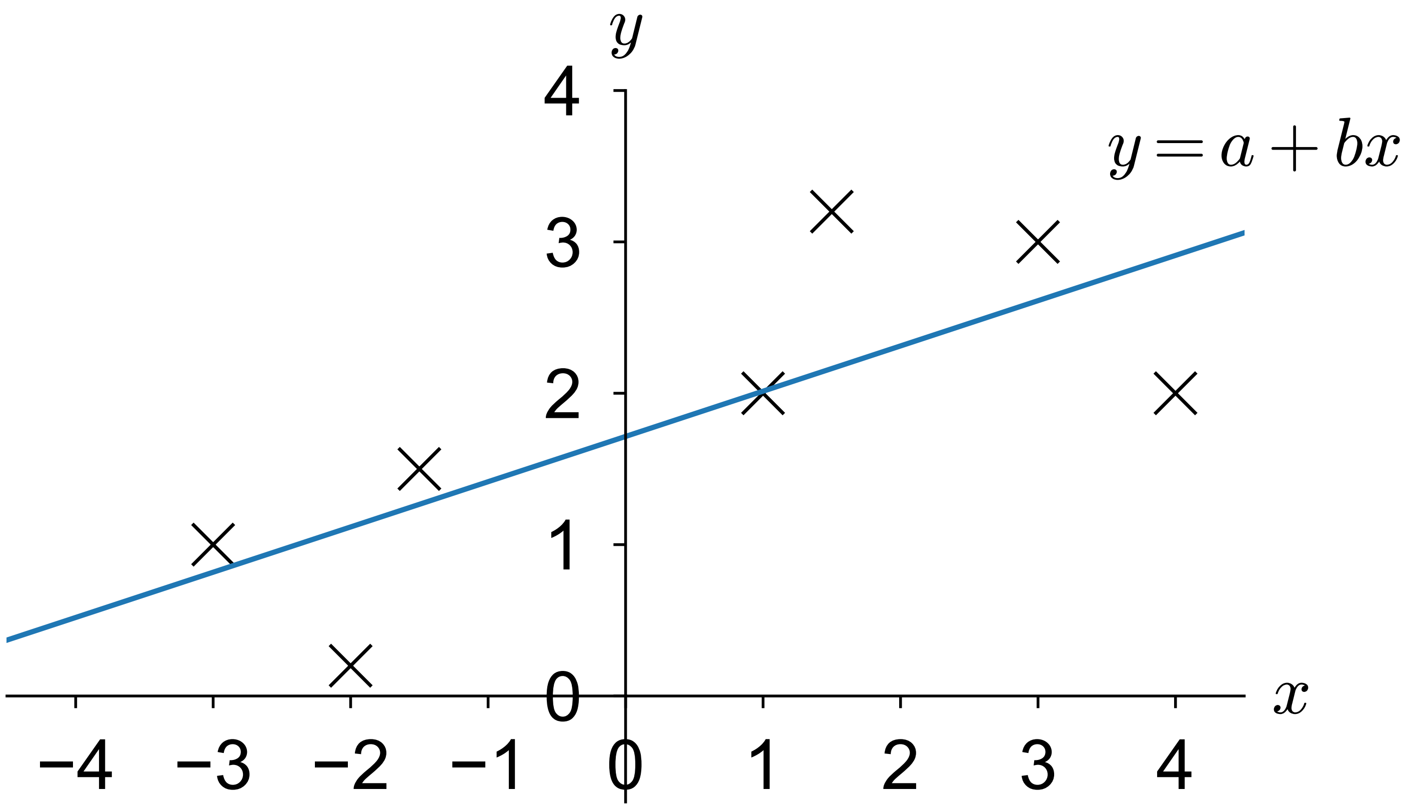
- Published in 2006
- For intermediate and expert users who need more mathematical constructions
- Textbook is downloadable for free: http://www.gaussianprocess.org/gpml/

Introductory textbook

- "Gaussian process and Machine Learning", Kodansha MLP series, 2019 in Japan
  - 55 Amazon Ratings so far
- From linear model to Gaussian processes
- Also covers unsupervised learning and variational inference
- Why deer?

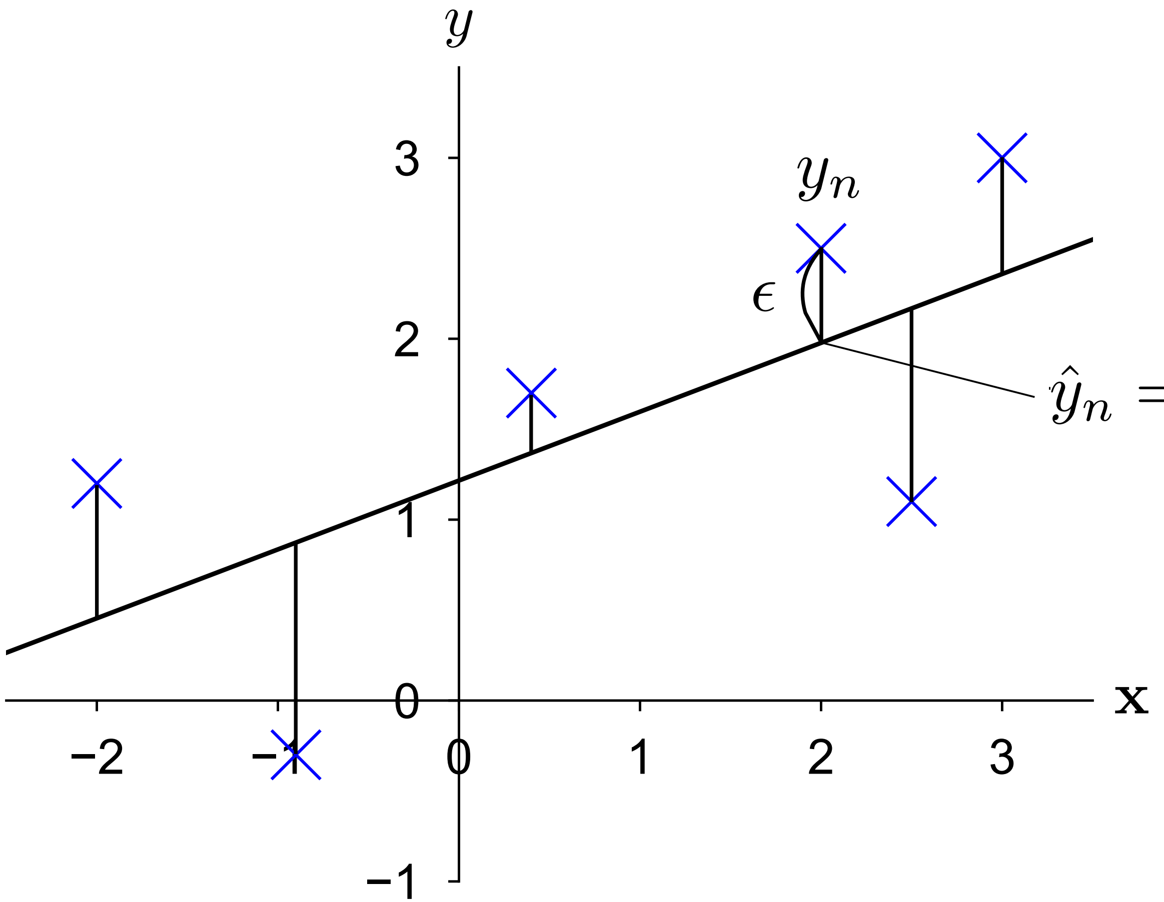
Simple regression

- Simplest prediction: $y = a + bx$
- How to determine *a* and *b*?



Minimization of error

- Minimize error ϵ between the observation y_n and prediction $\hat{y}_n = a + b\,x_n$: $\epsilon = y_n - \hat{y}_n = y_n - (a + bx_n)$



Simple regression (2)

- Given data $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$,
- Prediction $\hat{y}_n$ for each x_n is a linear function

$$\hat{y}_n = a + bx_n$$

- Difference between the observation is

$$y_n - \hat{y}_n = y_n - (a + bx_n)$$

  - We want to minimize this error!

Simple regression (3)

- For n=1,2,...,N,
  error= $y_n - \hat{y}_n$ → Minimize sum of errors
- Errors can be negative, thus take a square of errors (Least squares):

$$E = \sum_{n=1}^{N}(y_n - \hat{y}_n)^2 = \sum_{n=1}^{N}(y_n - (a + bx_n))^2$$

We want a,b to minimize this E

Simple regression

- Since derivative of E is zero at minimal point,

$$\frac{\partial E}{\partial a} = \frac{\partial}{\partial a}\sum_{n=1}^{N}(y_n - (a + bx_n))^2$$

$$= \frac{\partial}{\partial a}\sum_{n=1}^{N}(y_n^2 + a^2 + b^2x_n^2 - 2ay_n - 2abx_n + 2bx_ny_n) = 0$$

$$\frac{\partial E}{\partial b} = \frac{\partial}{\partial b}\sum_{n=1}^{N}(y_n - (a + bx_n))^2$$

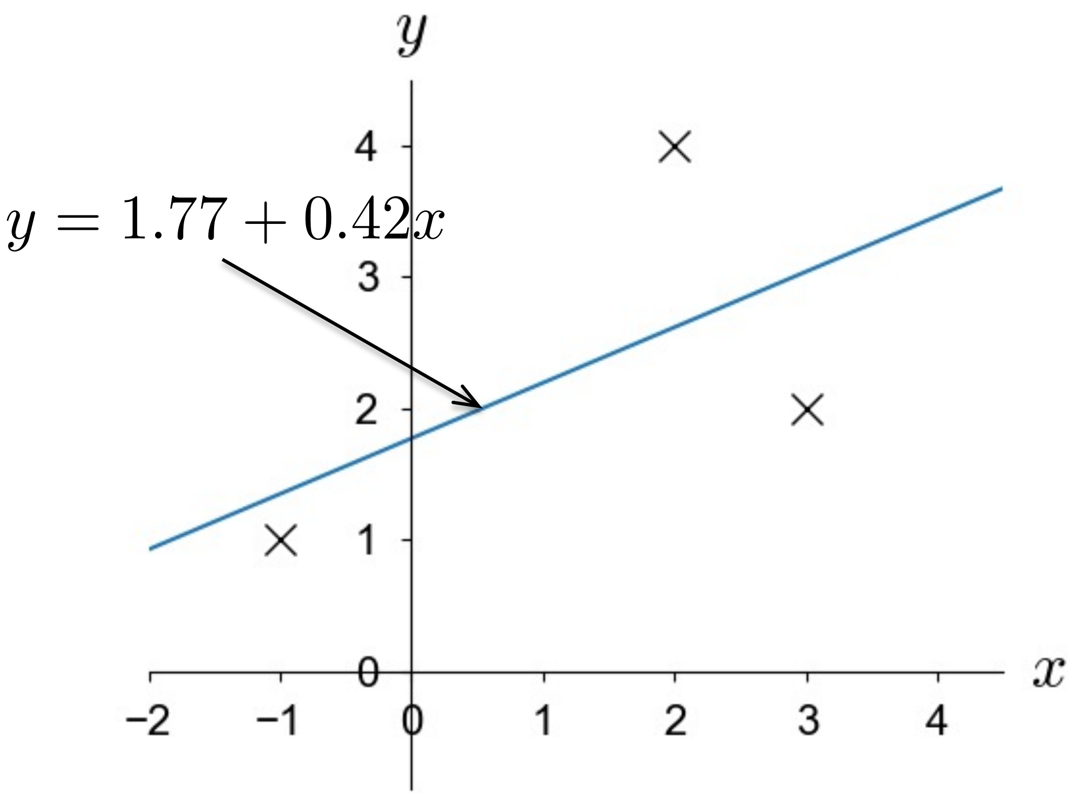
$$= \frac{\partial}{\partial b}\sum_{n=1}^{N}(y_n^2 + a^2 + b^2x_n^2 - 2ay_n - 2abx_n + 2bx_ny_n) = 0$$

- Yields

$$a = \frac{\sum_n x_n^2 \sum_n y_n - \sum_n x_n \sum_n x_n y_n}{N\sum_n x_n^2 - (\sum_n x_n)^2}$$

$$b = \frac{N\sum_n x_n y_n - \sum_n x_n \sum_n y_n}{N\sum_n x_n^2 - (\sum_n x_n)^2}$$

Simple regression (example)



Simplest case:
Data D=
{(3,2),(2,4),(-1,1)}

$$\sum_{n=1}^{3} x_n = 3 + 2 - 1 = 4$$

$$\sum_{n=1}^{3} y_n = 2 + 4 + 1 = 7$$

$$\sum_{n=1}^{3} x_n^2 = 9 + 4 + 1 = 14$$

$$\sum_{n=1}^{3} x_n y_n = 3 \cdot 2 + 2 \cdot 4 + (-1) \cdot 1 = 13$$

- Formula gives
  a = 1.77, b= 0.42

Multiple regression

- x is multi-dimensional? → Multiple regression

  When $\mathbf{x} = (x_1, x_2, \cdots, x_D)^T$,

  $y = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_D x_D$

- Squared error is

  $$(y - \hat{y})^2 = (y - (w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_D x_D))^2$$

- Minimize this error:
  → Take gradients of $E = \sum_{n=1}^{N} (y_n - \hat{y}_n)^2$ w.r.t. $w_0, w_1, \cdots, w_D$ to equate 0 and solve linear equations.

Vector-Matrix form

- When we set $\mathbf{x} = (1, x_1, x_2, \cdots, x_D)$ and $\mathbf{w} = (w_0, w_1, w_2, \cdots, w_D)$,

$$\begin{aligned}\hat{y} &= w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_D x_D \\ &= (w_0, w_1, w_2, \cdots, w_D) \begin{pmatrix} 1 \\ x_1 \\ \vdots \\ x_D \end{pmatrix} \\ &= \mathbf{w}^T \mathbf{x}\end{aligned}$$

Vector-Matrix form (2)

- Aligning vertically over n=1,2,...,N gives

$$\begin{pmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_N \end{pmatrix} = \begin{pmatrix} \mathbf{w}^T\mathbf{x}_1 \\ \mathbf{w}^T\mathbf{x}_2 \\ \vdots \\ \mathbf{w}^T\mathbf{x}_N \end{pmatrix} = \begin{pmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_N^T \end{pmatrix} \mathbf{w}$$

Called Design matrix

- Namely

$$\underbrace{\begin{pmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_N \end{pmatrix}}_{\hat{\mathbf{y}}} = \underbrace{\begin{pmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1D} \\ 1 & x_{21} & x_{22} & \cdots & x_{2D} \\ \vdots & & & & \vdots \\ 1 & x_{N1} & x_{N2} & \cdots & x_{ND} \end{pmatrix}}_{\mathbf{X}} \underbrace{\begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_D \end{pmatrix}}_{\mathbf{w}}$$

Can write $\hat{\mathbf{y}} = \mathbf{X}\mathbf{w}$

Vector-Matrix form (3)

$$E = \sum_{n=1}^{N} (y_n - \hat{y}_n)^2 = (y_1 - \hat{y}_1, \cdots, y_N - \hat{y}_N) \begin{pmatrix} y_1 - \hat{y}_1 \\ \vdots \\ y_N - \hat{y}_N \end{pmatrix}$$

- Therefore

$$\begin{aligned} E &= (\mathbf{y} - \mathbf{X}\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w}) \\ &= \mathbf{y}^T (\mathbf{y} - \mathbf{X}\mathbf{w}) - (\mathbf{X}\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w}) \\ &= \mathbf{y}^T \mathbf{y} - 2\mathbf{w}^T (\mathbf{X}^T \mathbf{y}) + \mathbf{w}^T \mathbf{X}^T \mathbf{X}\mathbf{w} \end{aligned}$$

Analytical solution of multiple regression

$$E = \mathbf{y}^T\mathbf{y} - 2\mathbf{w}^T(\mathbf{X}^T\mathbf{y}) + \mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w}$$

- taking derivative wrt $\mathbf{w}$,

$$\frac{\partial E}{\partial \mathbf{w}} = -2\mathbf{X}^T\mathbf{y} + 2\mathbf{X}^T\mathbf{X}\mathbf{w} = \mathbf{0}$$

- Thus

$$\mathbf{X}^T\mathbf{X}\mathbf{w} = \mathbf{X}^T\mathbf{y} \qquad \text{(Normal equation)}$$

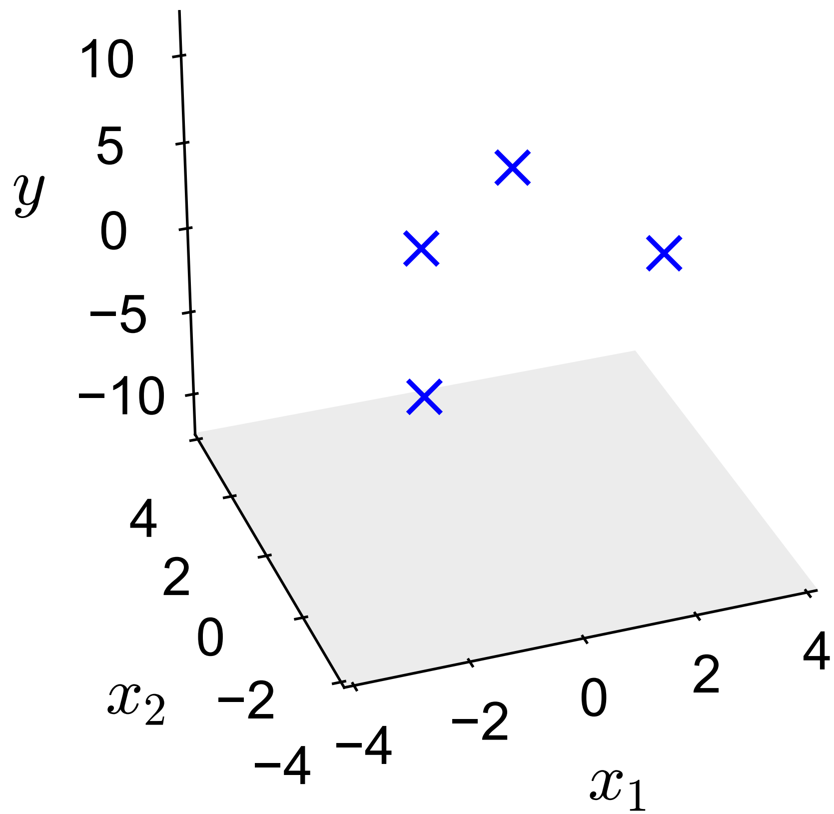
$$\therefore \quad \boxed{\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}\,.}$$

Solution of multiple regression

Multiple regression example

- Given data below:

$$\mathcal{D} = \{((1,2),4),((-1,1),2),((3,0),1),((-2,-2),-1)\}$$



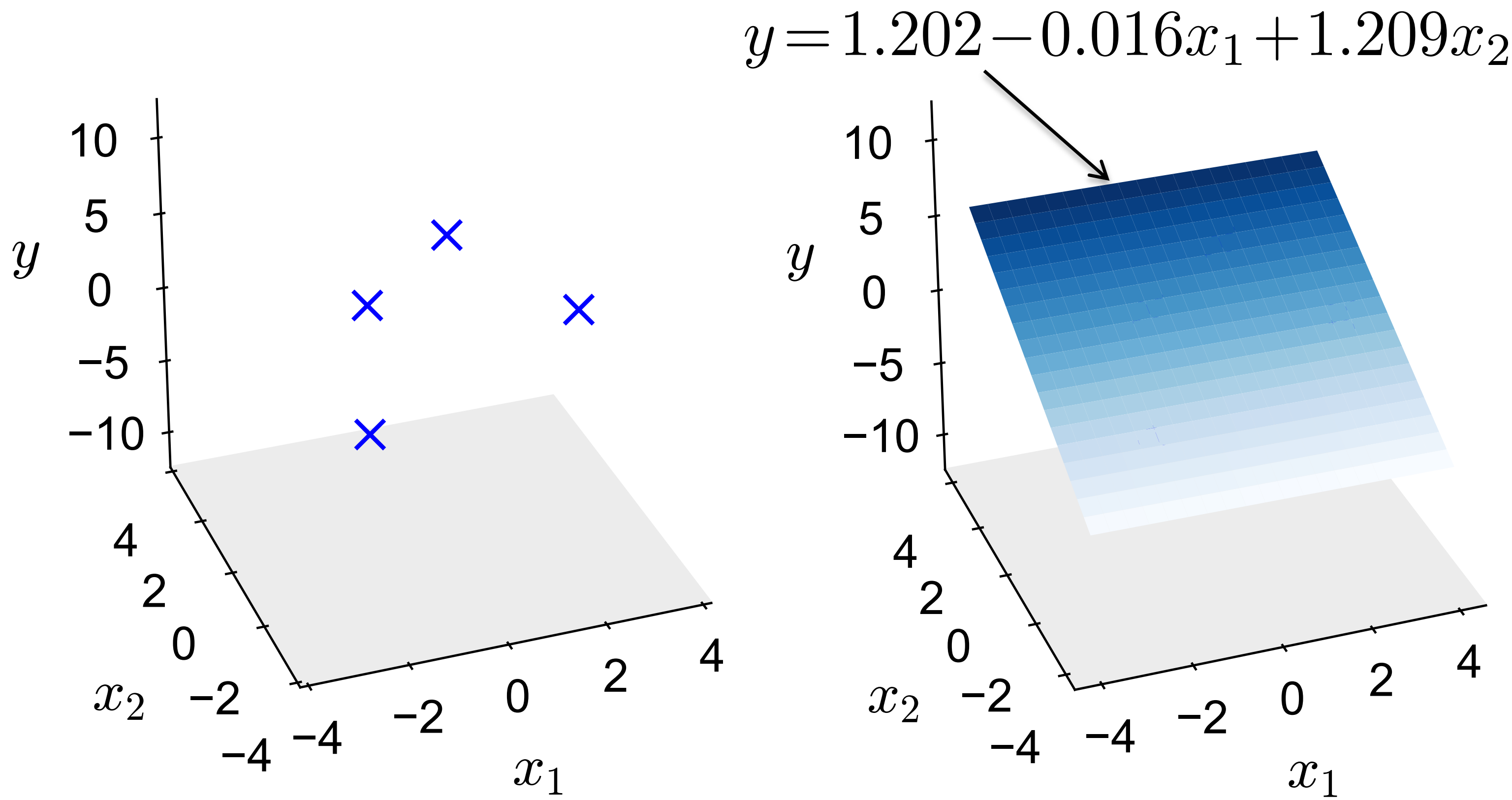
x_1	x_2	y
1	2	4
−1	1	2
3	0	1
−2	−2	−1

$$\mathbf{X} = \begin{pmatrix} 1 & 1 & 2 \\ 1 & -1 & 1 \\ 1 & 3 & 0 \\ 1 & -2 & -2 \end{pmatrix}$$

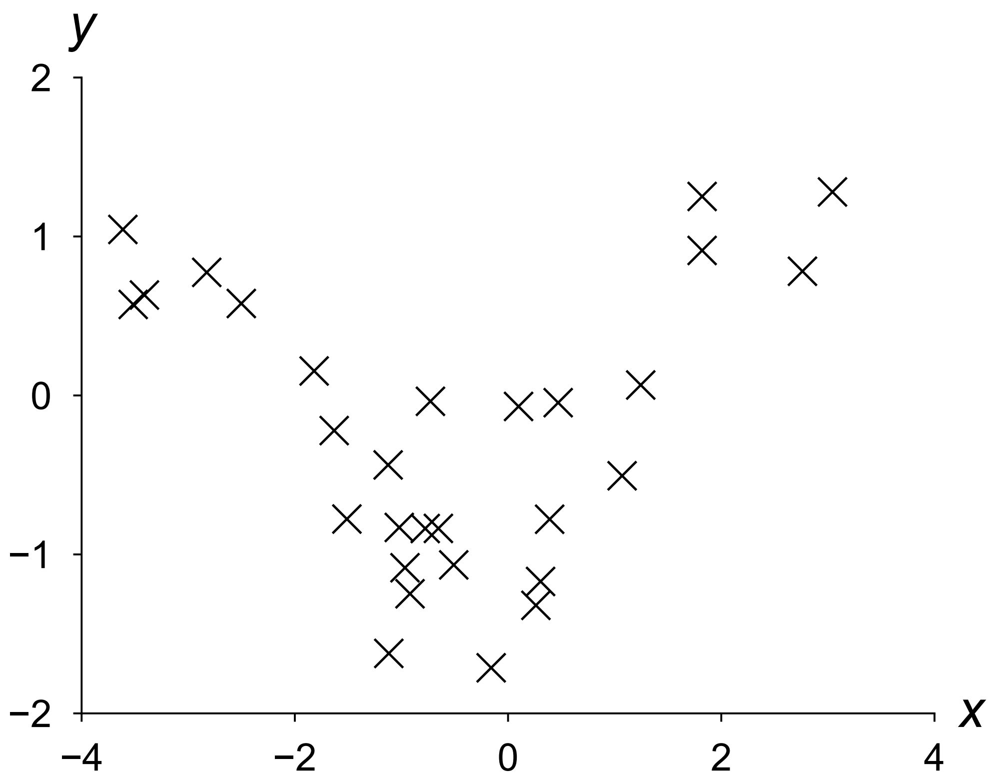
Multiple regression example (2)

- Therefore, optimal weight vector is

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} = (1.202 \;\; -0.016 \; 1.209)^T$$



More complex regression?



- Data often exhibit nonlinear relationships

More complex regression function!

Linear regression

$$y = w_0 + w_1 x + w_2 x^2 = \underbrace{\begin{pmatrix} w_0 & w_1 & w_2 \end{pmatrix}}_{\mathbf{w}^T} \underbrace{\begin{pmatrix} 1 \\ x \\ x^2 \end{pmatrix}}_{\phi(x)}$$

$$y = w_0 + w_1 x + w_2 \sin(x) = \underbrace{\begin{pmatrix} w_0 & w_1 & w_2 \end{pmatrix}}_{\mathbf{w}^T} \underbrace{\begin{pmatrix} 1 \\ x \\ \sin(x) \end{pmatrix}}_{\phi(x)}$$

- Both written as linear function of weights w
  - $y = \mathbf{w}^T \phi(\mathbf{x}) \cdots$ Linear regression
- Combining this with sigmoid function yields logistic regression:

$$y = \sigma(\mathbf{w}^T \phi(\mathbf{x}))$$

Linear regression

$$y = \mathbf{w}^T \phi(\mathbf{x}) + \epsilon$$

- For explanation, assume no noise:

$$y = \mathbf{w}^T \phi(\mathbf{x}) = w_1 \phi_1(\mathbf{x}) + w_2 \phi_2(\mathbf{x}) + \cdots + w_M \phi_M(\mathbf{x})$$

- For given pairs $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_N, y_N),\}$ this gives $\mathbf{y} = \mathbf{\Phi} \mathbf{w}$

$$\underbrace{\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix}}_{\mathbf{y}} = \underbrace{\begin{pmatrix} \phi_1(\mathbf{x}_1) & \phi_2(\mathbf{x}_1) & \cdots & \phi_M(\mathbf{x}_1) \\ \phi_1(\mathbf{x}_2) & \phi_2(\mathbf{x}_2) & \cdots & \phi_M(\mathbf{x}_2) \\ \vdots & & & \vdots \\ \phi_1(\mathbf{x}_N) & \phi_2(\mathbf{x}_N) & \cdots & \phi_M(\mathbf{x}_N) \end{pmatrix}}_{\mathbf{\Phi}} \underbrace{\begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_M \end{pmatrix}}_{\mathbf{w}}$$

From Linear regression to Gaussian processes

$$\underbrace{\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix}}_{\mathbf{y}} = \underbrace{\begin{pmatrix} \phi_1(\mathbf{x}_1) & \phi_2(\mathbf{x}_1) & \cdots & \phi_M(\mathbf{x}_1) \\ \phi_1(\mathbf{x}_2) & \phi_2(\mathbf{x}_2) & \cdots & \phi_M(\mathbf{x}_2) \\ \vdots & & & \vdots \\ \phi_1(\mathbf{x}_N) & \phi_2(\mathbf{x}_N) & \cdots & \phi_M(\mathbf{x}_N) \end{pmatrix}}_{\mathbf{\Phi}} \underbrace{\begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_M \end{pmatrix}}_{\mathbf{w}}$$

- When we assume $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \alpha\mathbf{I})$ like a ridge regression,
  $\mathbf{y} = \mathbf{\Phi}\mathbf{w}$ also obeys Gaussian distribution
  $$\mathbf{y} = \mathbf{\Phi}\mathbf{w} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$$
- What are $\boldsymbol{\mu}, \mathbf{\Sigma}$?

From Linear regression to Gaussian processes (2)

$$\mathbf{y} = \mathbf{\Phi}\mathbf{w} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$$

- $\boldsymbol{\mu} = \mathbb{E}[\mathbf{y}] = \mathbb{E}[\mathbf{\Phi}\mathbf{w}] = \mathbf{\Phi}\mathbb{E}[\mathbf{w}] = \mathbf{0}$
- $\mathbf{\Sigma} = \mathbb{E}[\mathbf{y}\mathbf{y}^T] - \mathbb{E}[\mathbf{y}]\mathbb{E}[\mathbf{y}]^T$
  $$= \mathbb{E}[(\mathbf{\Phi}\mathbf{w})(\mathbf{\Phi}\mathbf{w})^T] = \mathbb{E}[\mathbf{\Phi}\mathbf{w}\mathbf{w}^T\mathbf{\Phi}^T]$$
  $$= \alpha\mathbf{\Phi}\mathbf{\Phi}^T$$
- Therefore,
  $$\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \alpha\mathbf{\Phi}\mathbf{\Phi}^T)$$
  - This is called a Gaussian process!
    ($\mathbf{y}$ is jointly Gaussian)

From Linear regression to Gaussian processes (3)

$$\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \alpha \mathbf{\Phi}\mathbf{\Phi}^T) = \mathcal{N}(\mathbf{0}, \mathbf{K})$$

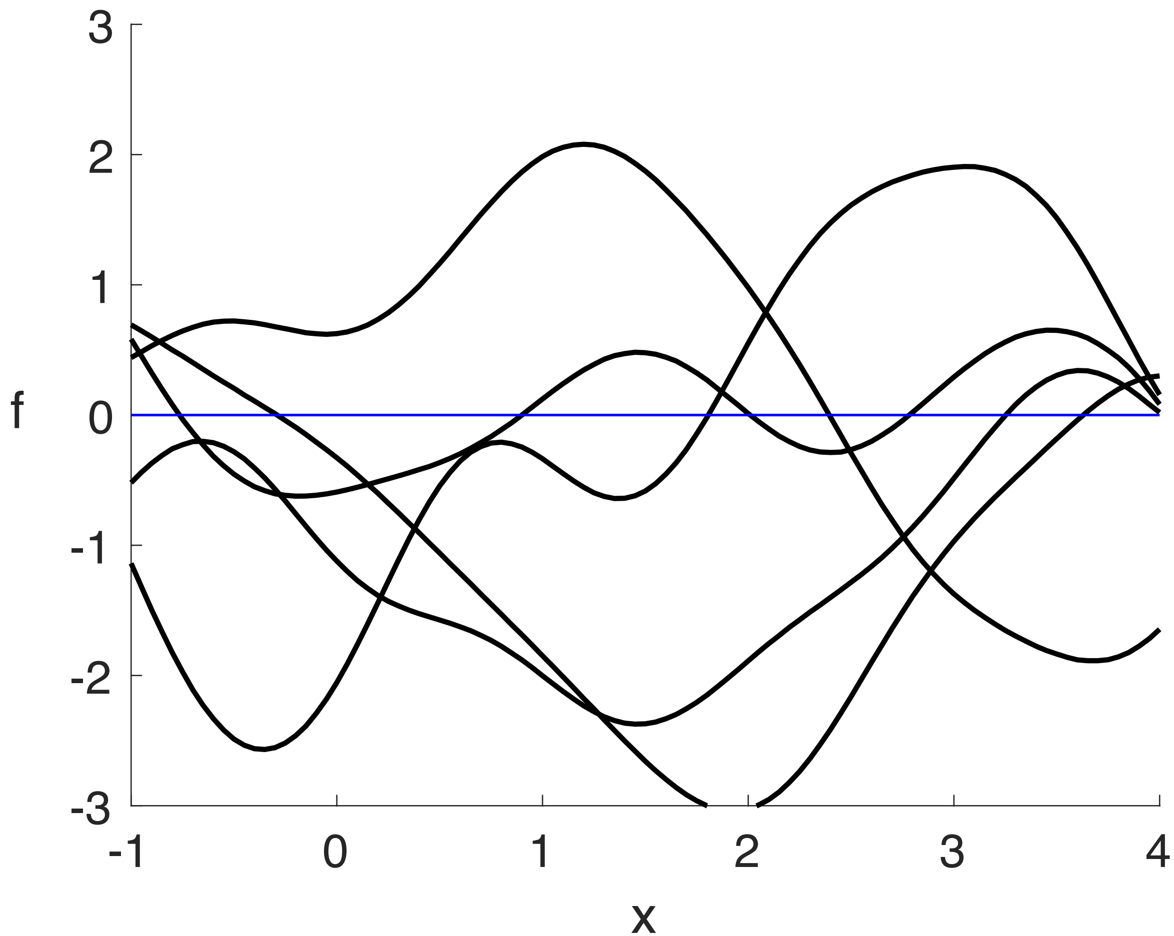
- Where

$$\mathbf{K} = \alpha \mathbf{\Phi}\mathbf{\Phi}^T = \alpha \underbrace{\begin{pmatrix} \vdots \\ \boxed{\phi(\mathbf{x}_i)^T} \\ \vdots \end{pmatrix}}_{\mathbf{\Phi}} \underbrace{\begin{pmatrix} \cdots & \boxed{\phi(\mathbf{x}_j)} & \cdots \end{pmatrix}}_{\mathbf{\Phi}^T}$$

$$K_{ij} = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) = k(\mathbf{x}_i, \mathbf{x}_j)$$: kernel function

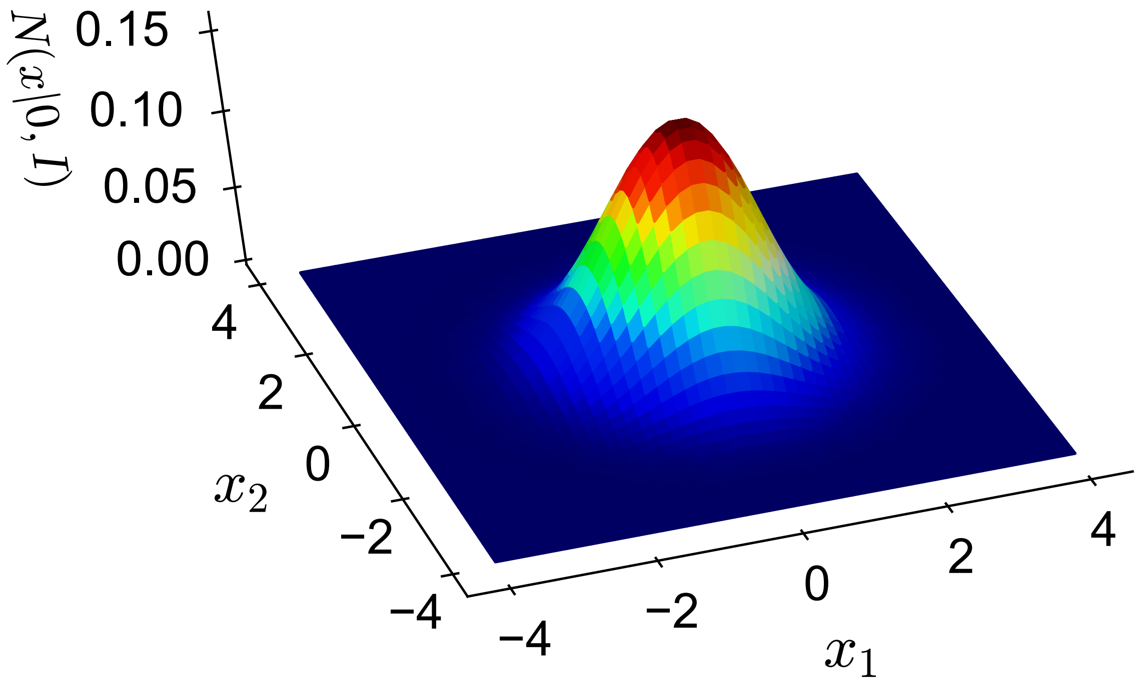
- Now, $\mathbf{w}$ are integrated out in a Bayesian way

Random draw from a Gaussian process



- Why such smooth functions?

Multivariate Gaussian distribution

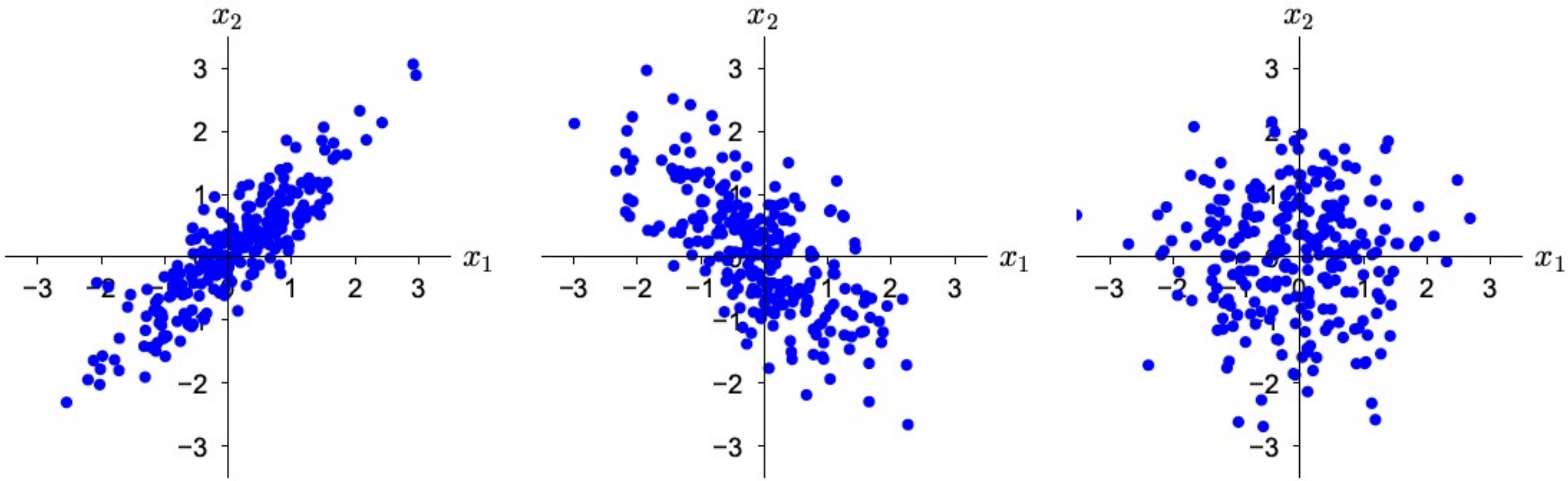


- $$p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(\sqrt{2\pi})^D \sqrt{|\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}-\boldsymbol{\mu}) \right)$$

$\boldsymbol{\mu} = \mathbb{E}[\mathbf{x}]$ (mean vector)

$\boldsymbol{\Sigma} = \mathbb{E}[\mathbf{x}\mathbf{x}^T] - \mathbb{E}[\mathbf{x}]\mathbb{E}[\mathbf{x}]^T$ (covariance matrix)

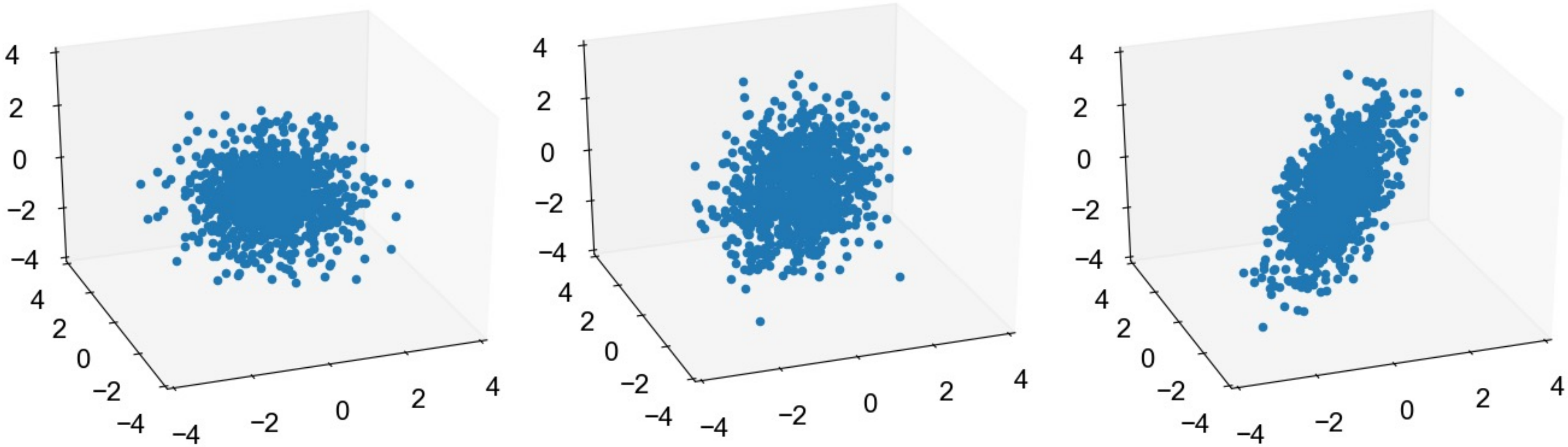
Random draws from MVN



(a) $\boldsymbol{\Sigma} = \begin{pmatrix} 1 & 0.9 \\ 0.9 & 1 \end{pmatrix}$ (b) $\boldsymbol{\Sigma} = \begin{pmatrix} 1 & -0.7 \\ -0.7 & 1 \end{pmatrix}$ (c) $\boldsymbol{\Sigma} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

- If correlation is high, similar values are sampled
  - Minus for negative correlation, 0 for independent

Random draws from MVN (2)



$$\boldsymbol{\Sigma} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad \boldsymbol{\Sigma} = \begin{pmatrix} 1 & 0.5 & 0.2 \\ 0.5 & 1 & 0.5 \\ 0.2 & 0.5 & 1 \end{pmatrix} \qquad \boldsymbol{\Sigma} = \begin{pmatrix} 1 & 0.8 & 0.6 \\ 0.8 & 1 & 0.8 \\ 0.6 & 0.8 & 1 \end{pmatrix}$$

- Three dimensional case

Intuitive explanation

- Assume grid of input points

$$\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N)$$

and create a covariance matrix $\mathbf{K}$ between them:

$$K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$$

e.g.

$$\mathbf{K} = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{cccc} \quad x_1 & \quad\quad x_2 & \quad\quad x_3 & \quad\quad x_4 \end{array} \\ \begin{pmatrix} 1.0000 & 0.3679 & 0.0183 & 0.0001 \\ 0.3679 & 1.0000 & 0.3679 & 0.0183 \\ 0.0183 & 0.3679 & 1.0000 & 0.3679 \\ 0.0001 & 0.0183 & 0.3679 & 1.0000 \end{pmatrix} \end{array}$$

kernel function

- $k(x_i,x_j)$: kernel function. For example:

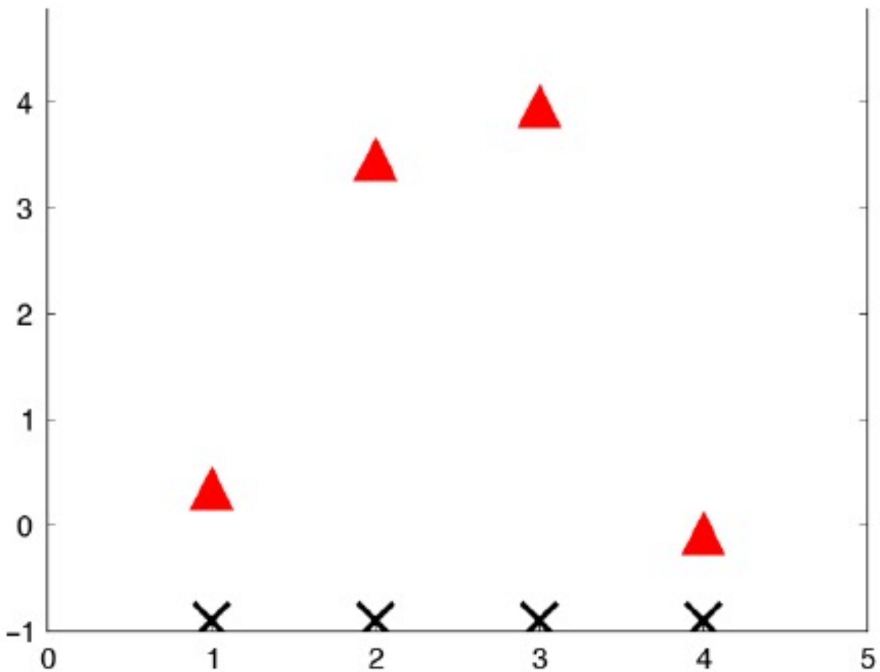
$$k(\mathbf{x}, \mathbf{x}') = \exp(-|\mathbf{x} - \mathbf{x}'|^2/\theta)$$

(RBF kernel)

- Same as those used in SVM or others
(GP is a Bayesian version of the kernel methods)

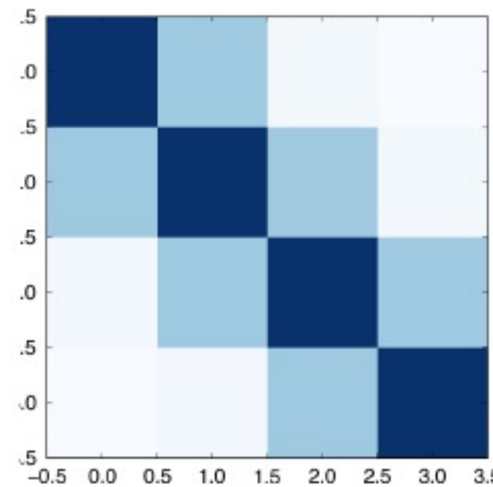
Intuitive explanation (2)

- Draw from correlated multivariate Gaussian:



Sample from MVN

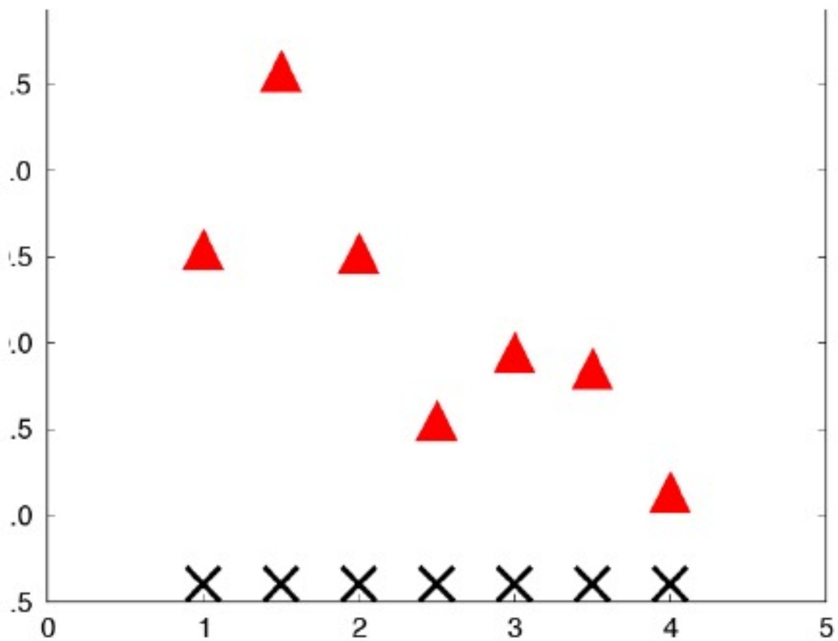
$K =$



Covariance matrix

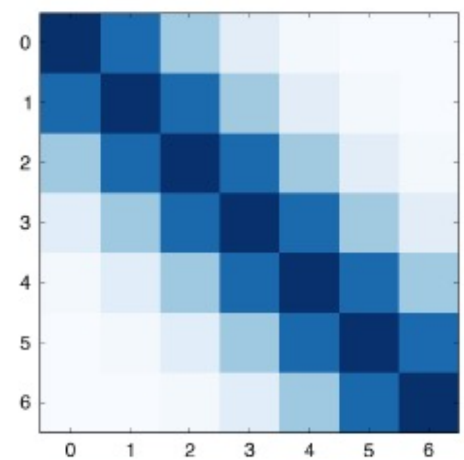
Intuitive explanation (3)

- Draw from correlated multivariate Gaussian:



Sample from MVN

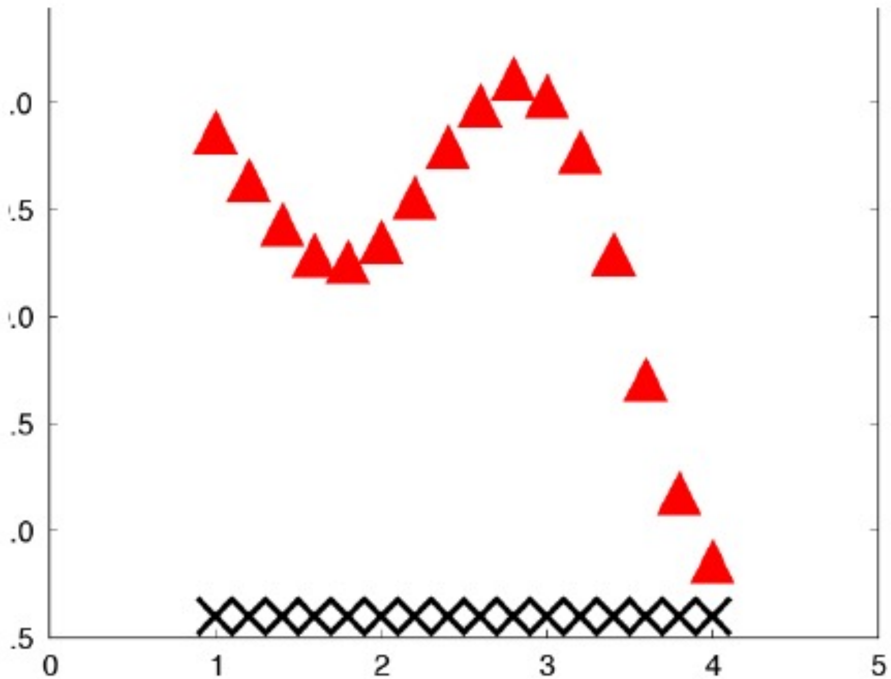
$K =$



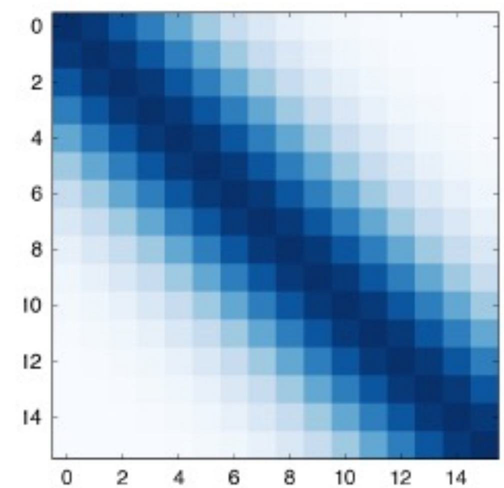
Covariance matrix

Intuitive explanation (4)

- Draw from correlated multivariate Gaussian:

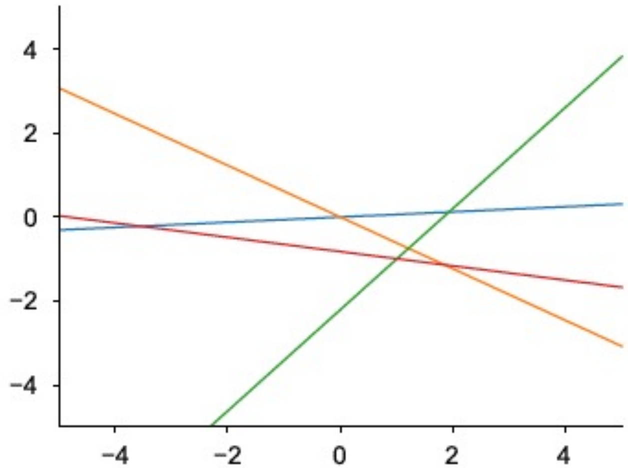


Sample from MVN

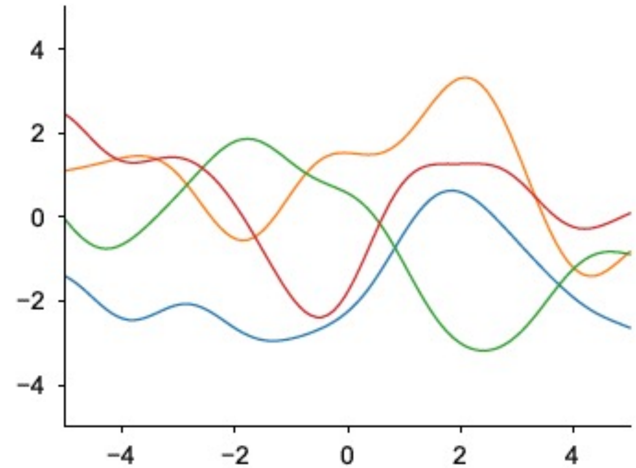
$K =$ 

Covariance matrix

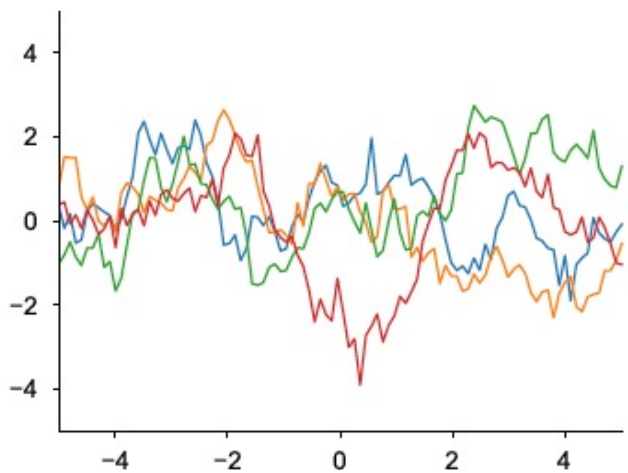
Kernels and random draws from GP



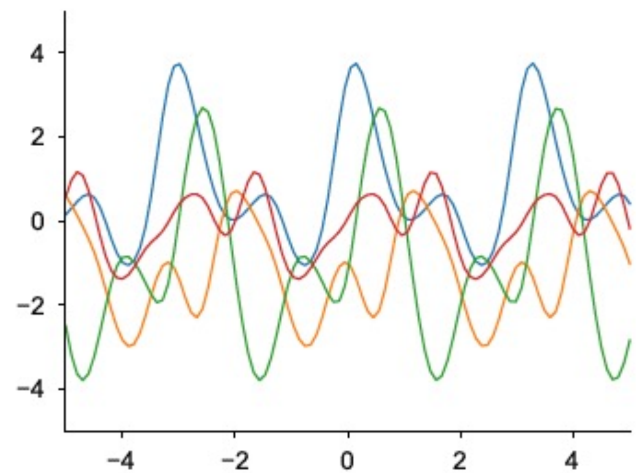
(a) Linear kernel: $\mathbf{x}^T\mathbf{x}'$



(b) RBF kernel: $\exp(-|\mathbf{x}-\mathbf{x}'|^2/\theta)$



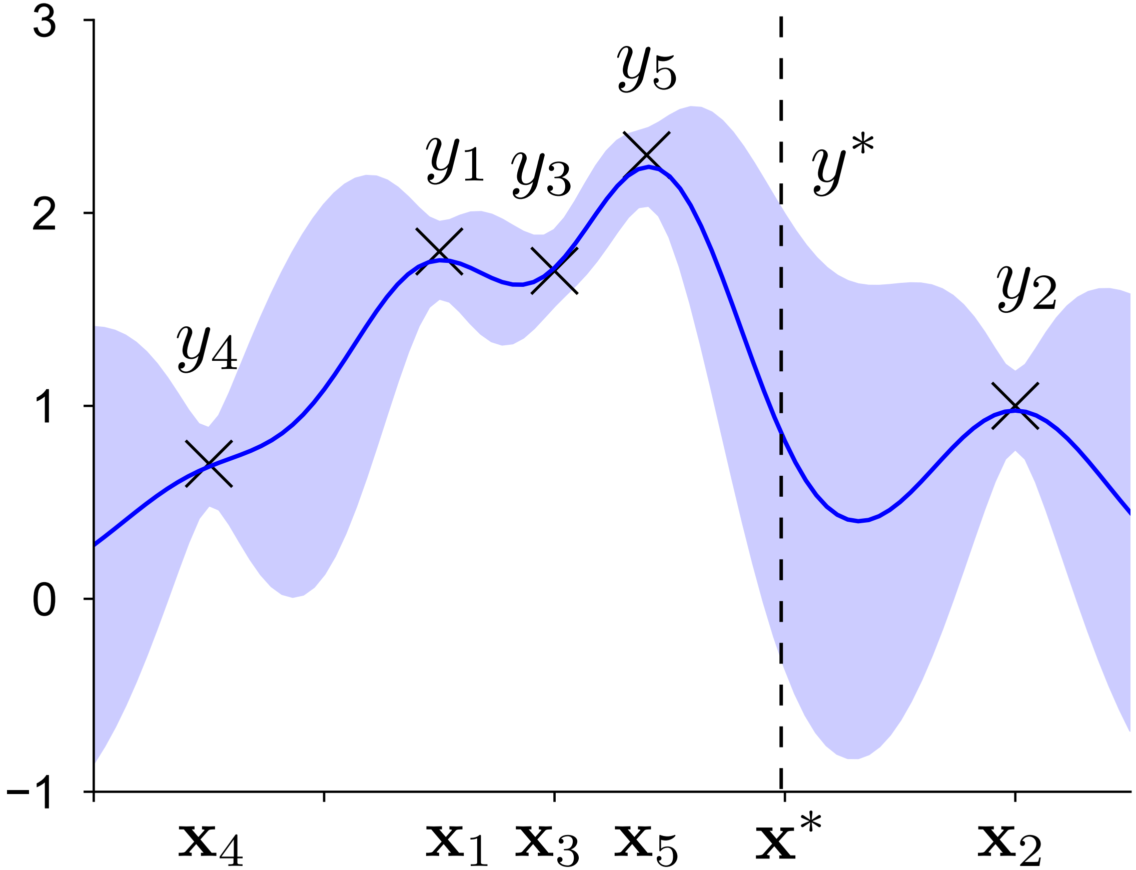
(c) Exponential kernel: $\exp(-|\mathbf{x}-\mathbf{x}'|/\theta)$



(d) Periodic kernel: $\exp(\cos(|\mathbf{x}-\mathbf{x}'|/\theta))$

Ordinary usage: GPR

- Gaussian process regression (GPR)
  - Given a set of pairs (*x*,*y*), predict y for new x*



Part 2:
Language, Robotics, and Gaussian processes

Language and Robotics

- (Intellectual) Robotics is not only around physical movements, but symbolic actions are necessary for high-level behavior
  - Closely tied with languages

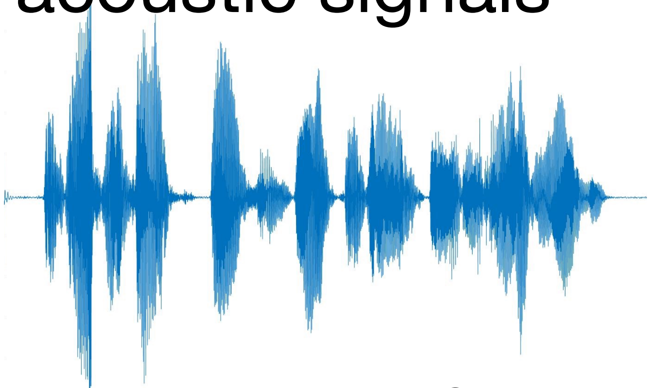
What is “word”?

- Children are not taught about “words” : learn only from experiences

- Robots should be flexible enough to learn “words” automatically from their environment
- Novel words, colloquial expressions, dialects, and child languages .. should be robust in a practical viewpoint

What is “word”?

- Problem: want to find “words” from a sequence of acoustic signals



→ “she sings the song”

- Easier version: find words from phonological sequences of child’s speech (Goldwater+ 2006)

WAtsDIs → WAts DIs
WAtIzIt → WAt Iz It
yuwanttusiD6bUk → yu want tu si D6 bUk

(CHILDES corpus)

Word segmentation

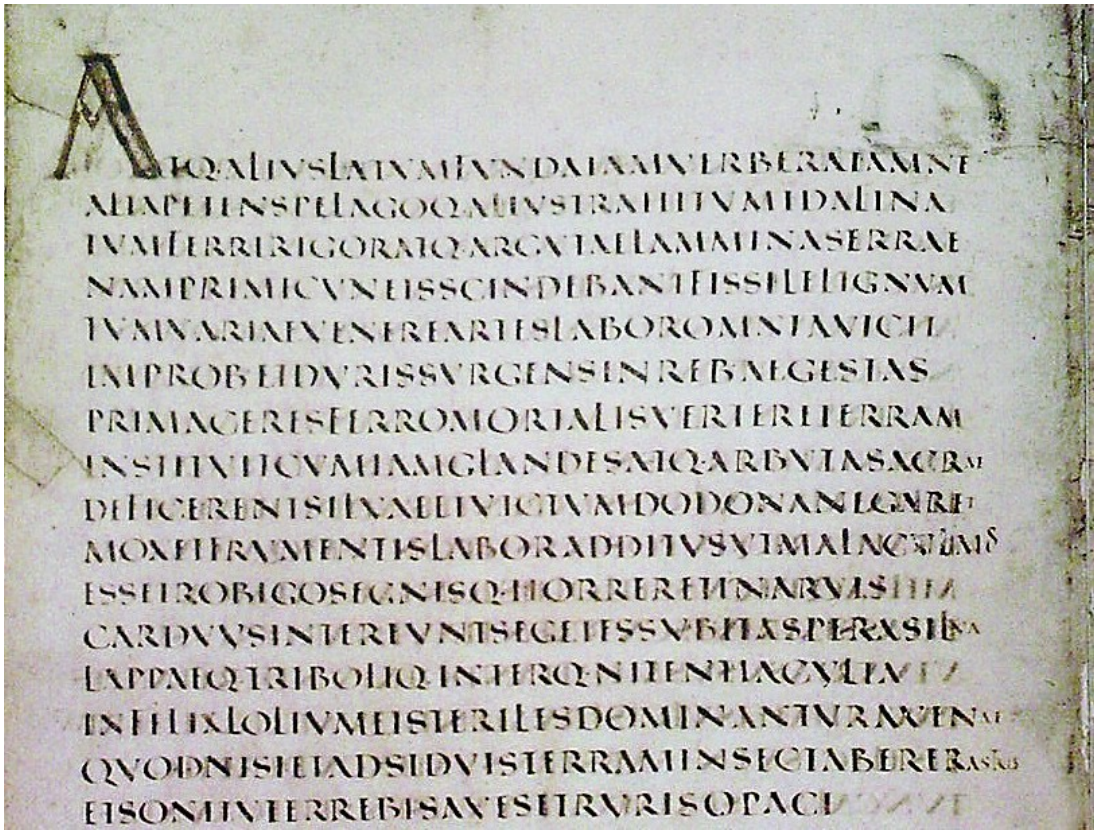
- Word segmentation and morpheme analysis are crucial tool to analyze Asian languages

山花貞夫・新民連会長は十六日の記者会見で
→ 山花 貞夫 ・ 新 民連 会長 は 十六 日 の 記者 会見 で
今后一段时期，不但居民会更多地选择国债
→ 今后 一段 时期 ， 不但 居民 会 更 多 地 选择 国债

- Usually done by supervised learning, but can be made unsupervised (Mochihashi+ ACL 2009, Uchiumi+ ACL 2015)!

Western languages

- Case of Latin (*Scripta continua*)
  - English was also originally written without spaces



Vergil text, around AD 141

Unsupervised word segmentation

(Mochihashi+ ACL 2009)

first,shedreamedoflittlealiceherself,andonceagainthetinyhandswereclaspedup
onherknee,andthebrighteagereyeswerelookingupintohersshecouldheartheevery
tonesofhervoice,andseethatqueerlittletossofherheadtokeepbackthewanderingh
airthatwouldalwaysgetintohereyesandstillasshelistened,orseemedtolisten,thew
holeplacearoundherbecamealivethestrangecreaturesofherlittlesister'sdream.the
longgrassrustledatherfeetasthewhiterabbithurriedbythefrightenedmousesplashe
dhiswaythroughtheneighbouringpoolshecouldheartherattleoftheteacupsasthema

first, she dream ed of little alice herself ,and once again the tiny hand s were clasped upon her knee ,and the bright eager eyes were looking up into hers -- shecould hearthe very tone s of her voice , and see that queer little toss of herhead to keep back the wandering hair that would always get into hereyes -- and still as she listened , or seemed to listen , thewhole place a round her became alive the strange creatures of her little sister 'sdream. thelong grass rustled ather feet as thewhitera bbit hurried by -- the frightened mouse splashed his way through the neighbour ing pool -- shecould hearthe rattle ofthe tea cups…

Arabic unsupervised segmentation

- Arabic AFP news, 40,000 sentences

الفلسطينيبسببتظاهرةلانصارحركةالمقاومةالاسلاميةحماس.

واذاتحققذلكفانكيسلوفسكييكونقدحازثلاث ... كبرنفيابرزثلاثة

... صحية

قائد+ ...

... مالايقل

وقالتدانييلتومسونالتيكتبتالسيناريو ... استغرقاعدادهخمسةاعوام."تاريخي

NPY.LM

Google translate:
"Filstinebsbptazahrplansarhrkpalmquaompalaslamiphamas."

الفلسطيني بسبب تظاهرة ل انصار حركة المقاومة الاسلامية حماس .

و اذا تحقق ذلك ف ان كيسلوفسكي يكون قد حاز ثلاث ... كبرنفيابرز ثلاثة

... الصحية

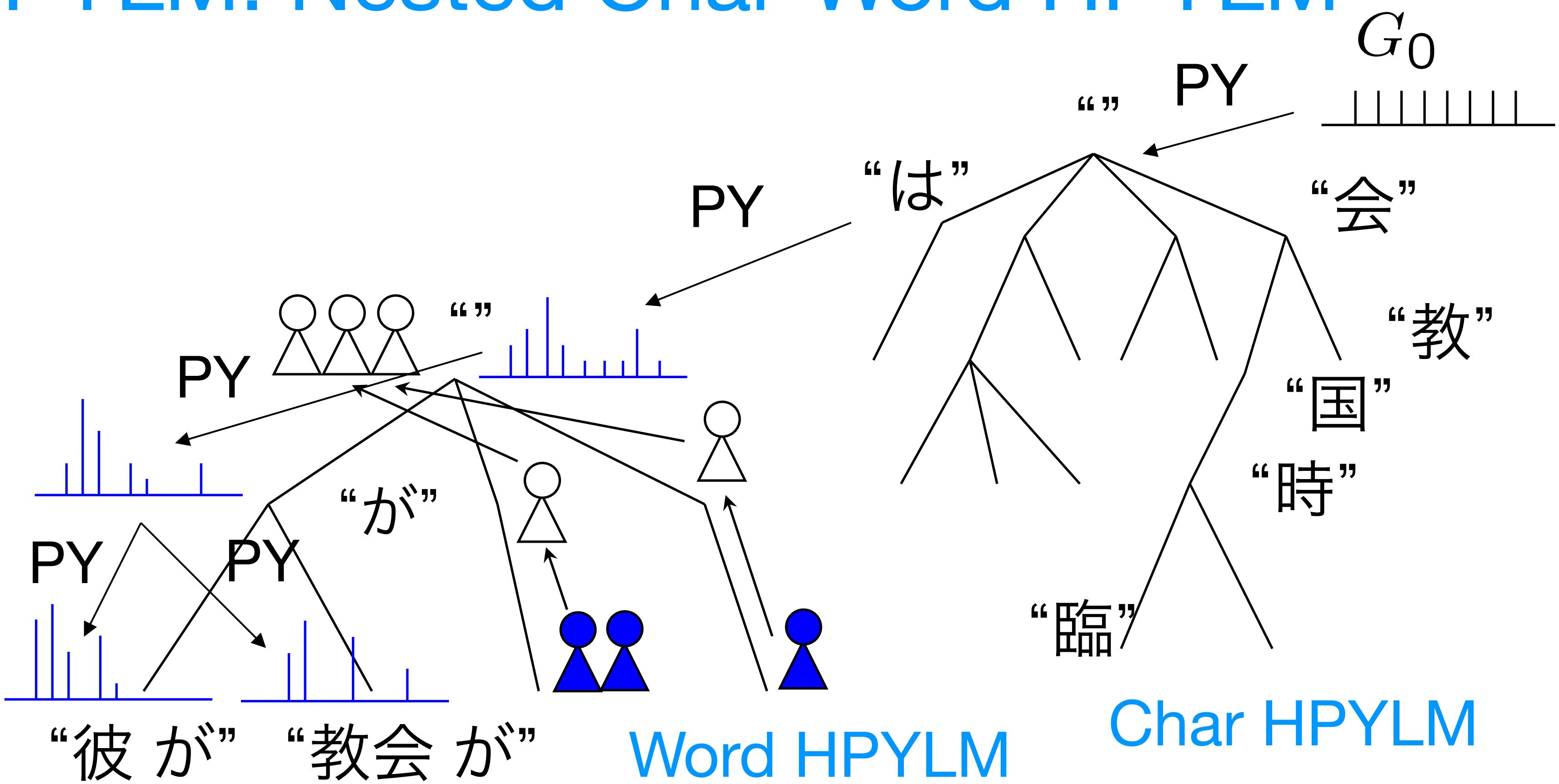
... سطينية

... مالايقل

... تاريخي

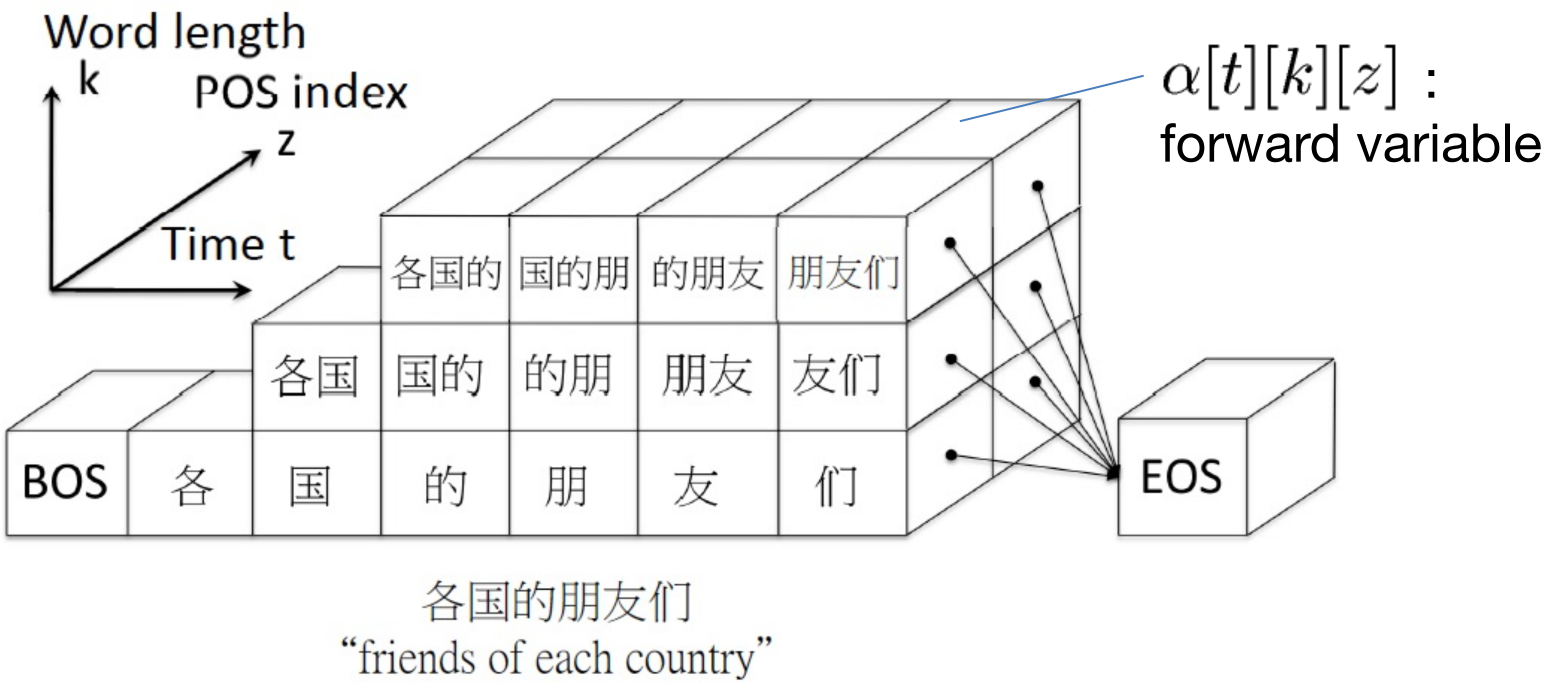
Google translate:
"Palestinian supporters of the event because of the Islamic Resistance Movement, Hamas."

NPYLM: Nested Char-Word HPYLM



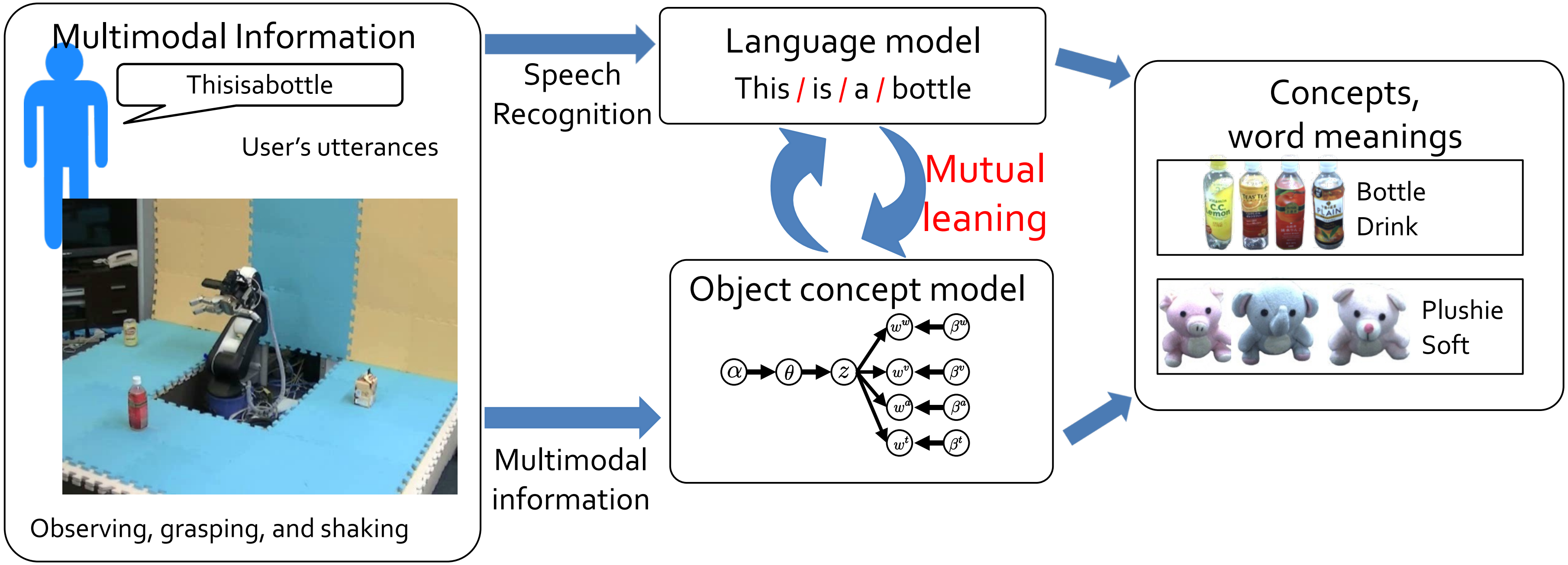
- Nested Character-Word n-gram model (HPYLM)
  - i.e. Bayesian hierarchical Markov model

Inference with Dynamic Programming



- "Inducing Word and Part-of-speech with Pitman-Yor Hidden Semi-Markov models", ACL 2015
- Forward filtering-backward sampling MCMC to escape from local minima

Learning with Multimodal information



"*Mutual Learning of an Object Concept and Language Model based on MLDA and NPYLM*",

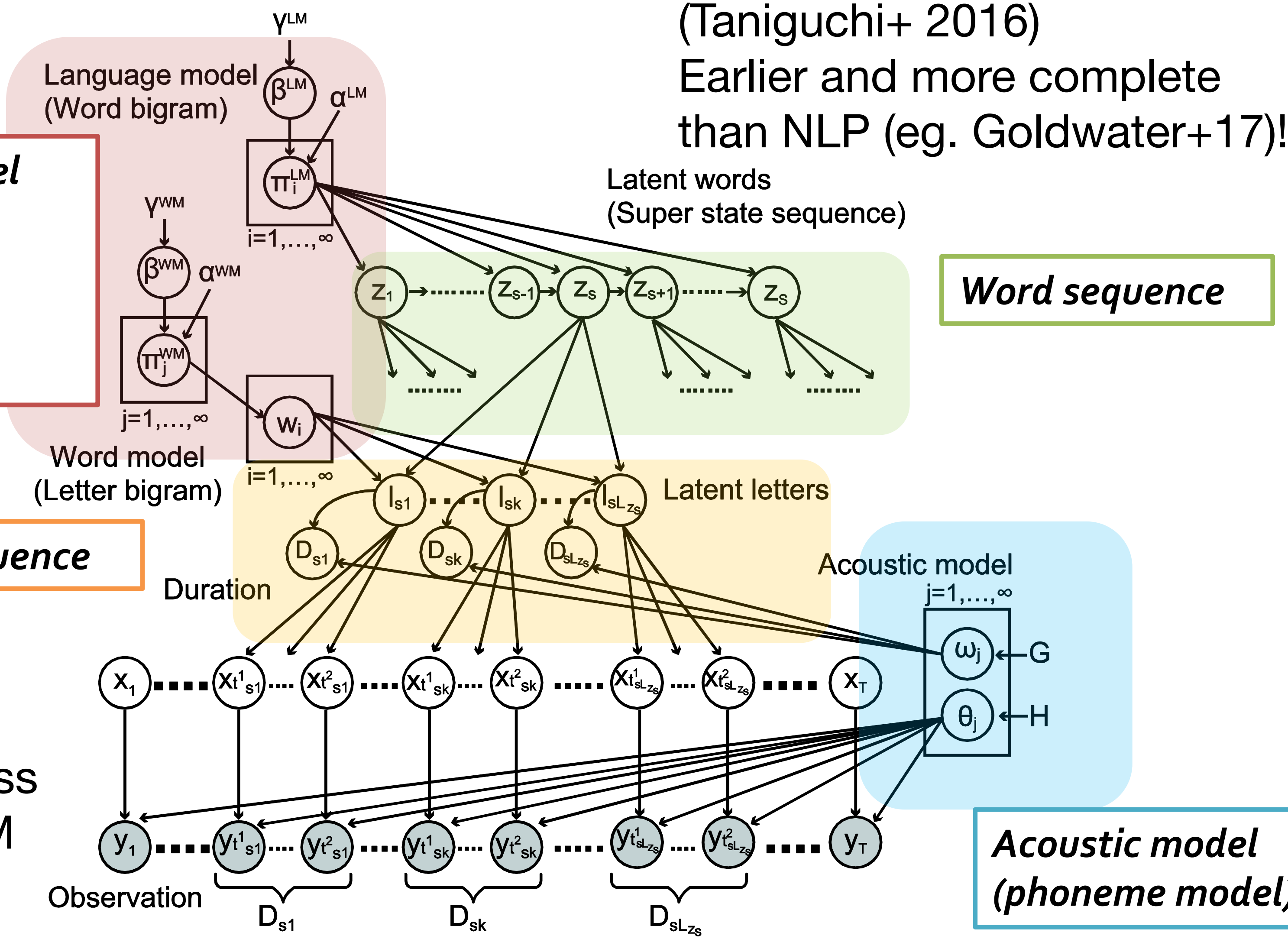
Nakamura et al., IROS 2014

Word discovery from acoustic signals

(Taniguchi+ 2016)
Earlier and more complete than NLP (eg. Goldwater+17)!

Language model (Word bigram model with letter bigram model)

Phoneme sequence



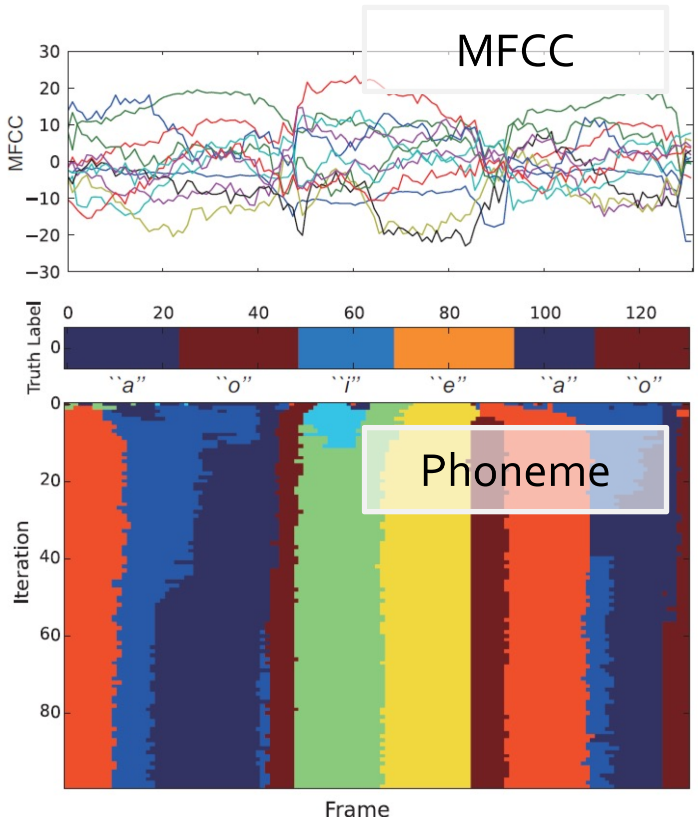
HDP-HLM:
Hierarchical
Dirichlet process
Hierarchical LM

Word discovery from acoustic signals (2)

- ✓ Five artificial words {aioi, aue, ao, ie, uo} prepared by connecting five Japanese vowels.
- ✓ 30 sentences (25 two-word and 5 three-word sentences) are prepared and each sentence is recorded twice by four Japanese speakers.
- ✓ MFCC (frame size =25ms, shift = 10ms, frame rate 100hz)

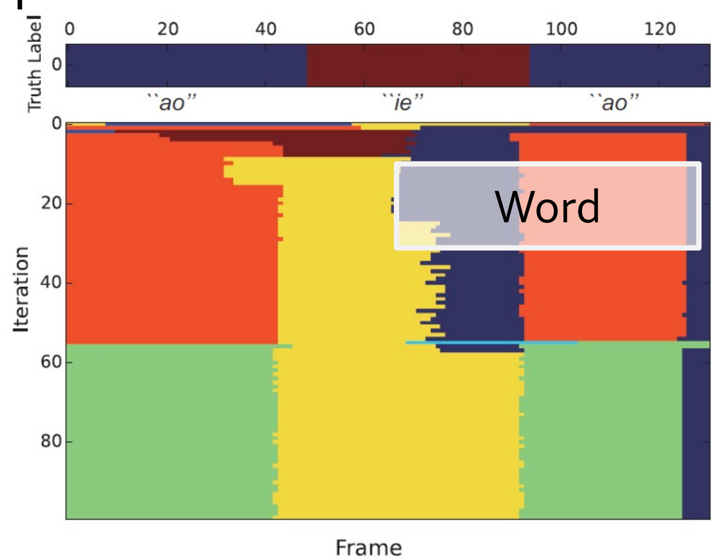
uo aue ie
ie ie uo
aue ao ie
ao ie ao
aioi uo ie

* HDP-HLM are trained separately for each speaker.



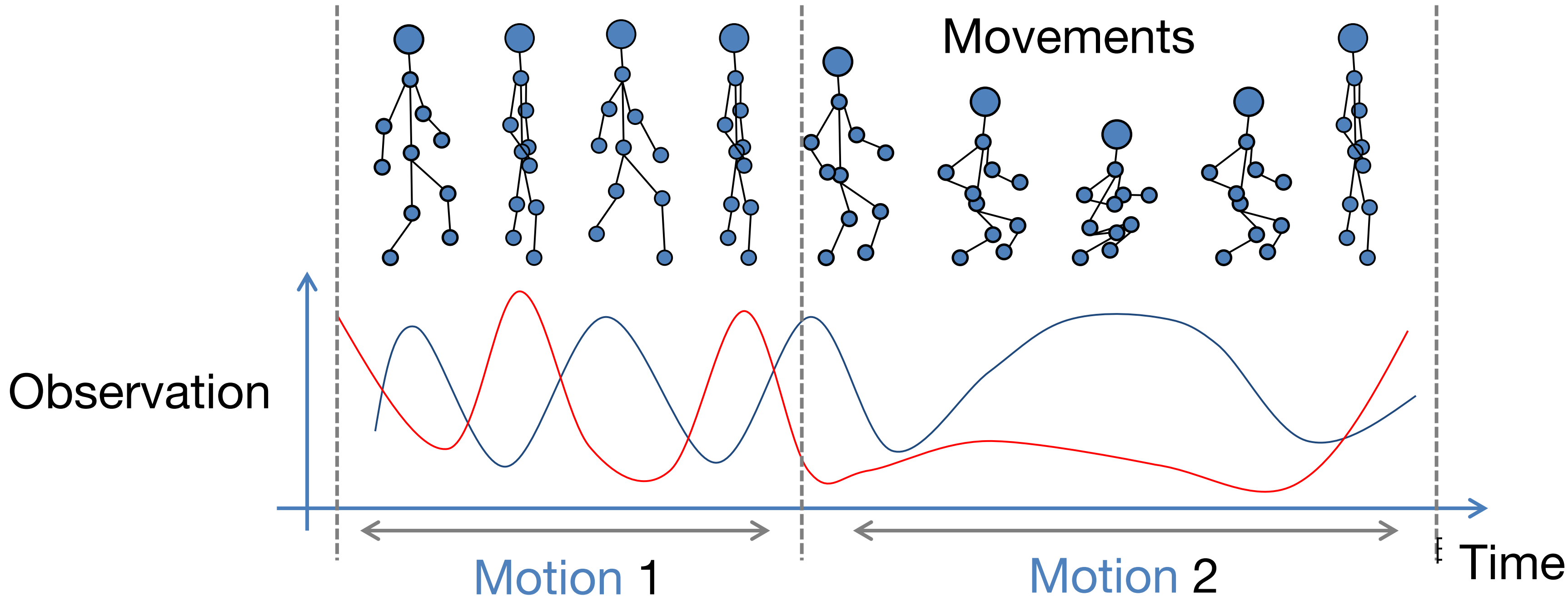
- ✓ The inference procedure could gradually estimate the boundaries of words and phonemes.

ex) ao-ie-ao



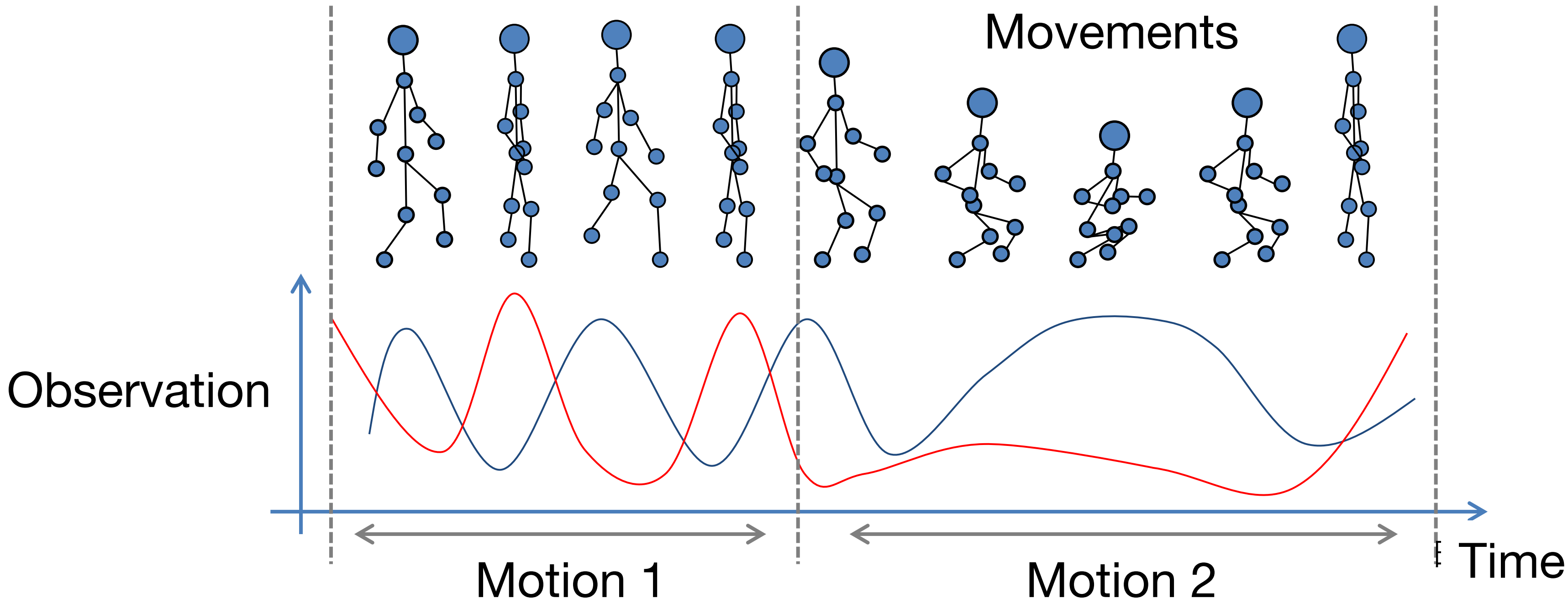
Part 2:
Motion segmentation and Gaussian processes

Learning “Motion” from Movements



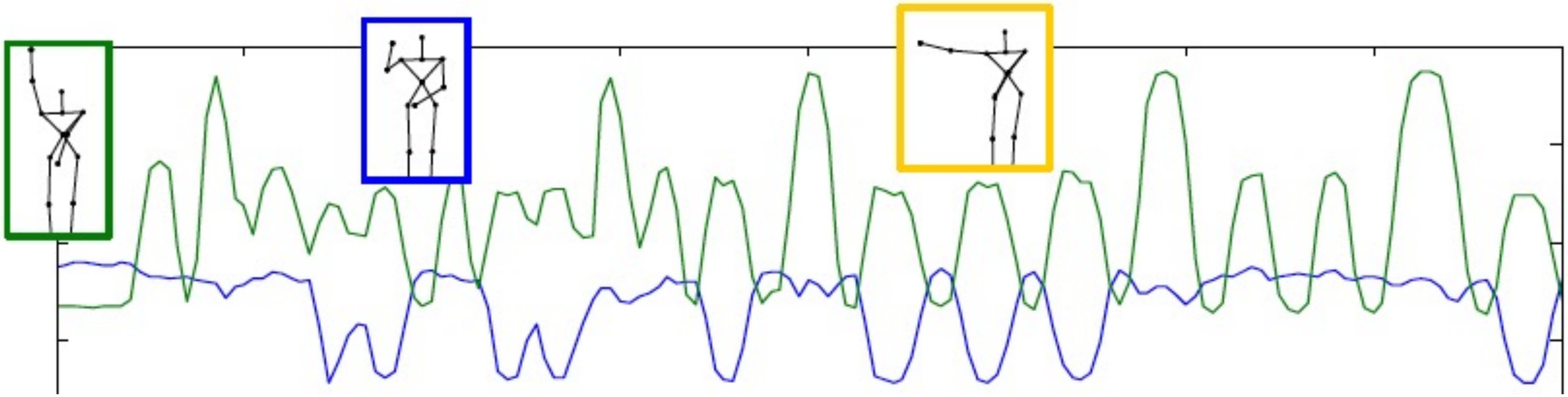
- Crucial for high-level recognition and planning of actions of robots .. word segmentation for robots
  - Planning for preparing dishes: wash->cut->boil->...

Model of movements



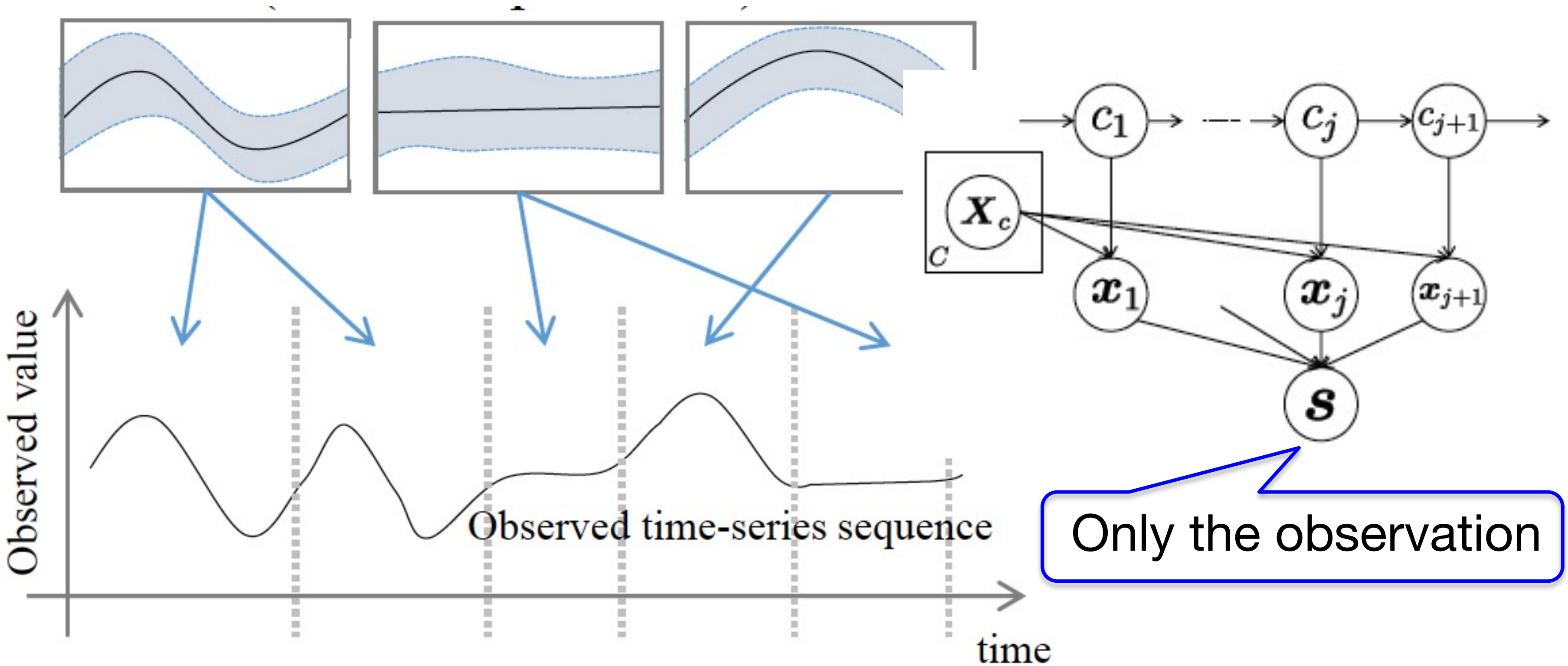
- Observations are continuous→ Stochastic models for trajectories are necessary
- We leverage the Gaussian processes

Actual data of movements



- Time series from two angles (knee and shoulder)
- How to induce "motion" from this data?

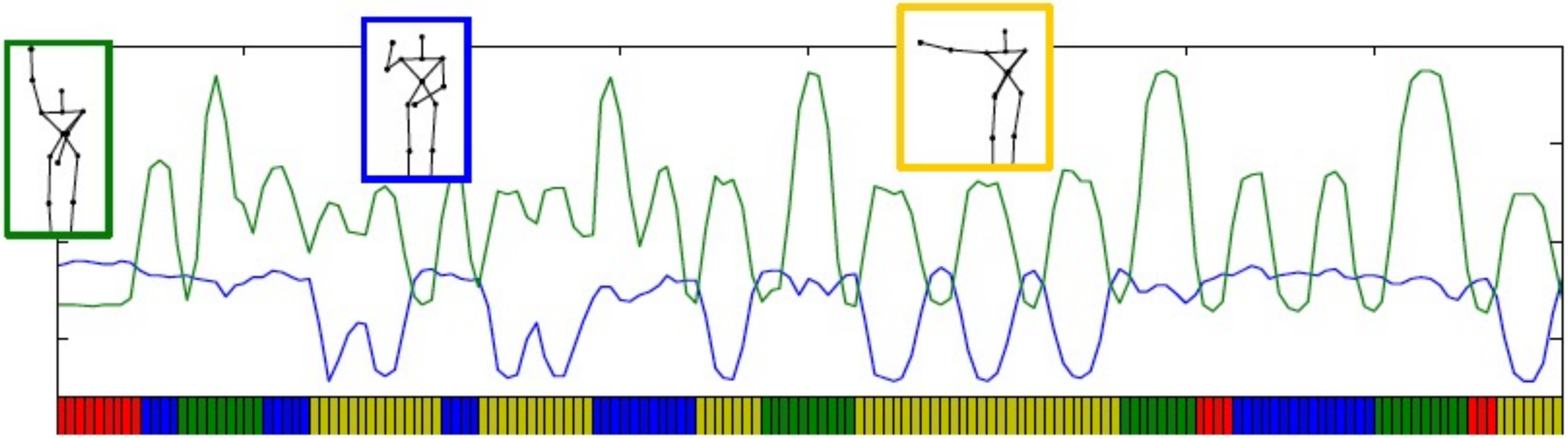
Gaussian process hidden semi-Markov model



- Hidden state (motion class): $c_j \sim \mathrm{Mult}(c|c_{j-1})$
- Generate trajectory: $\mathbf{x}_j \sim \mathrm{GP}(\mathbf{x} \,|\, \mathbf{X}_{c_j})$
- a kind of HMM, but segmentations are unknown

Application to Robotics

- Hidden semi-Markov model with Gaussian process observations to model the time series of joint angles, Forward-Backward Bayesian learning (infer “words”)
- Segmentation results using simple arm motions :



Segmentation result

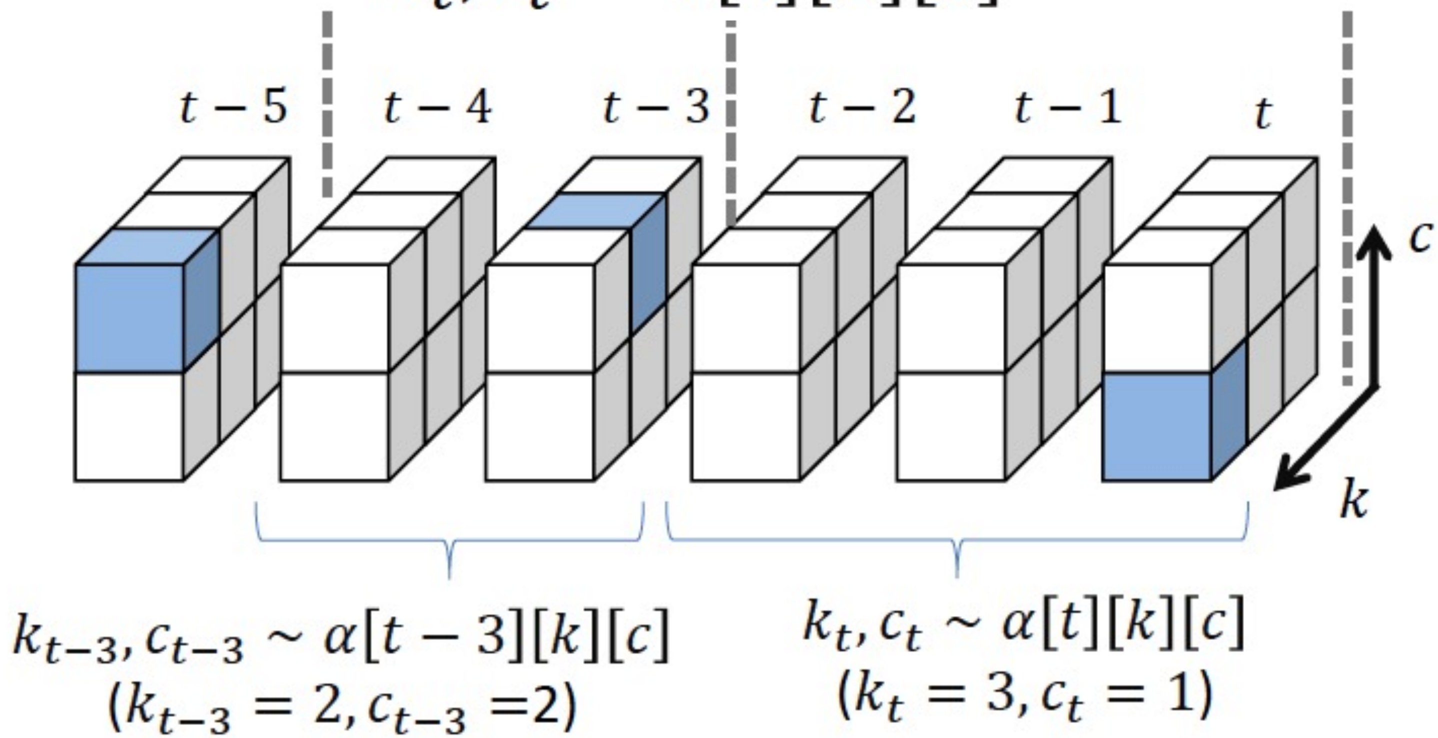
Dynamic programming MCMC

- Forward filtering:

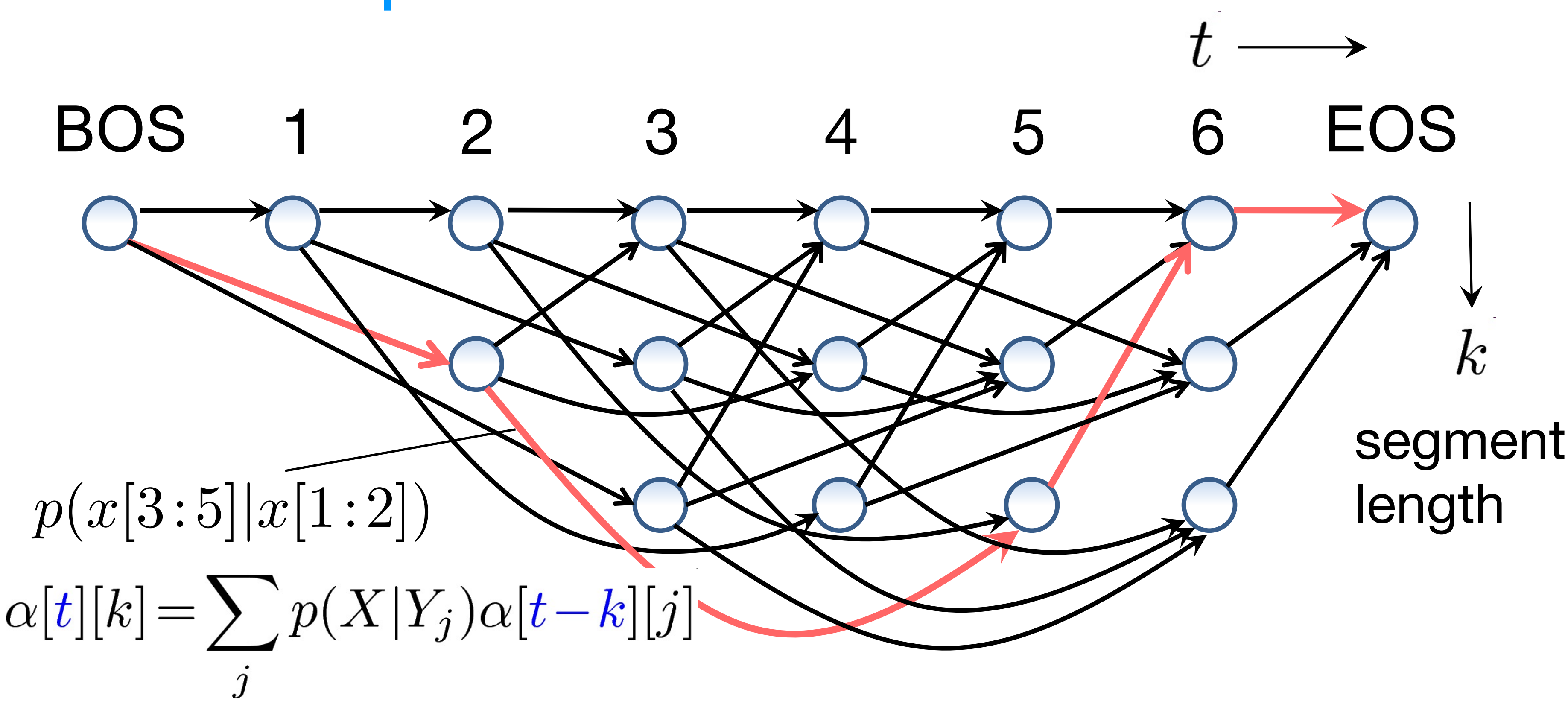
$$\alpha[t][k][c] = GP(\boldsymbol{s}_{t-k:k}|\boldsymbol{X}_c)\sum_{k}\sum_{c} p(c|c')\alpha[t-k][k'][c']$$

- Backward sampling:

$$k_t, c_t \sim \alpha[t][k][c]$$



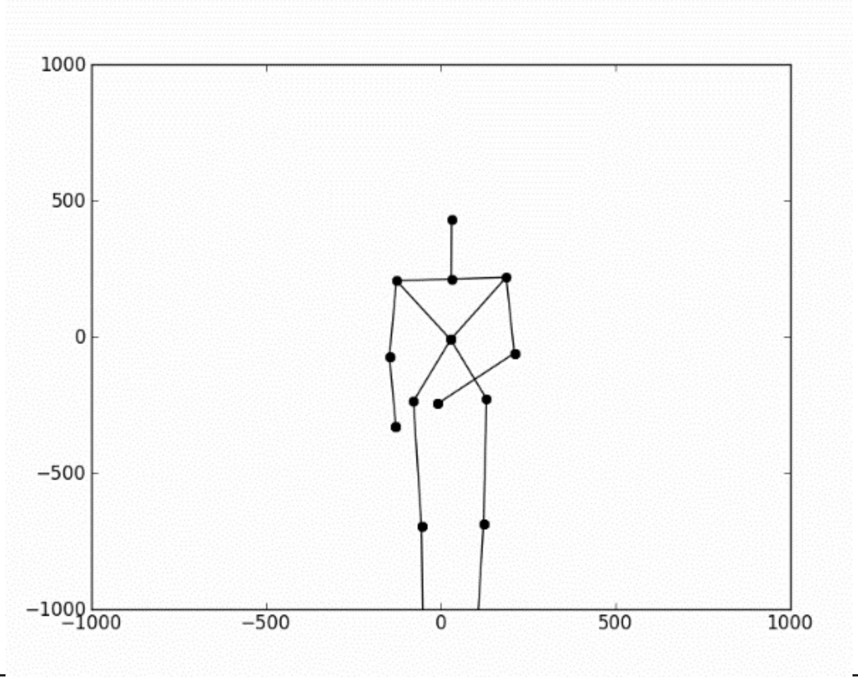
Lattice representation



- Semi-Markov HMM (Murphy 02, Ostendorf 96)
  - We do not know “correct path” beforehand (as HMM)
- Draw high-probability path via dynamic programming

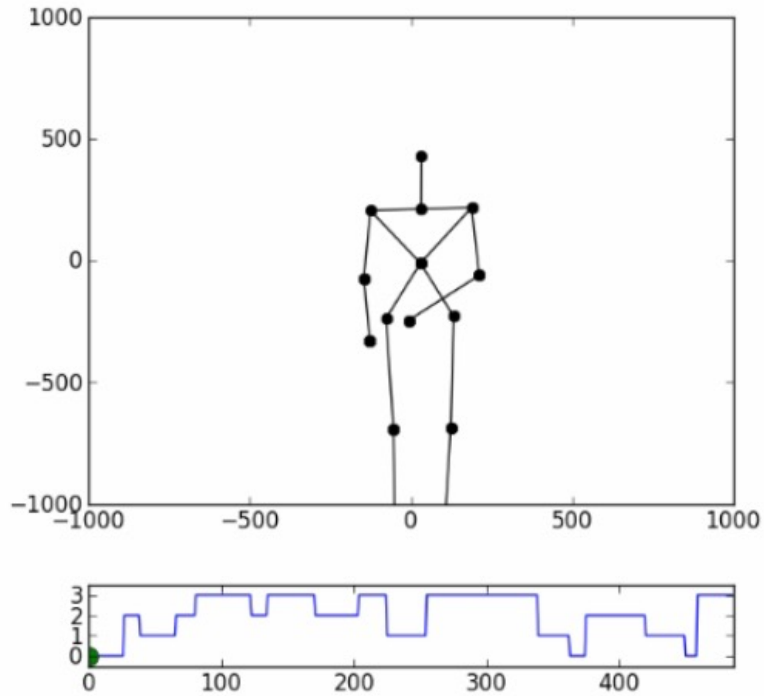
The first experiment

- Unsupervised segmentation of arm movements measured by Kinect
- 2D coordinates (x,y) of the right hand
  - Assume x and y are independently generated: $x \sim GP(c)$, $y \sim GP(c)$
- Motions involved:
  - Move the hand to right
  - Raise the hand high
  - Raise the hand slightly

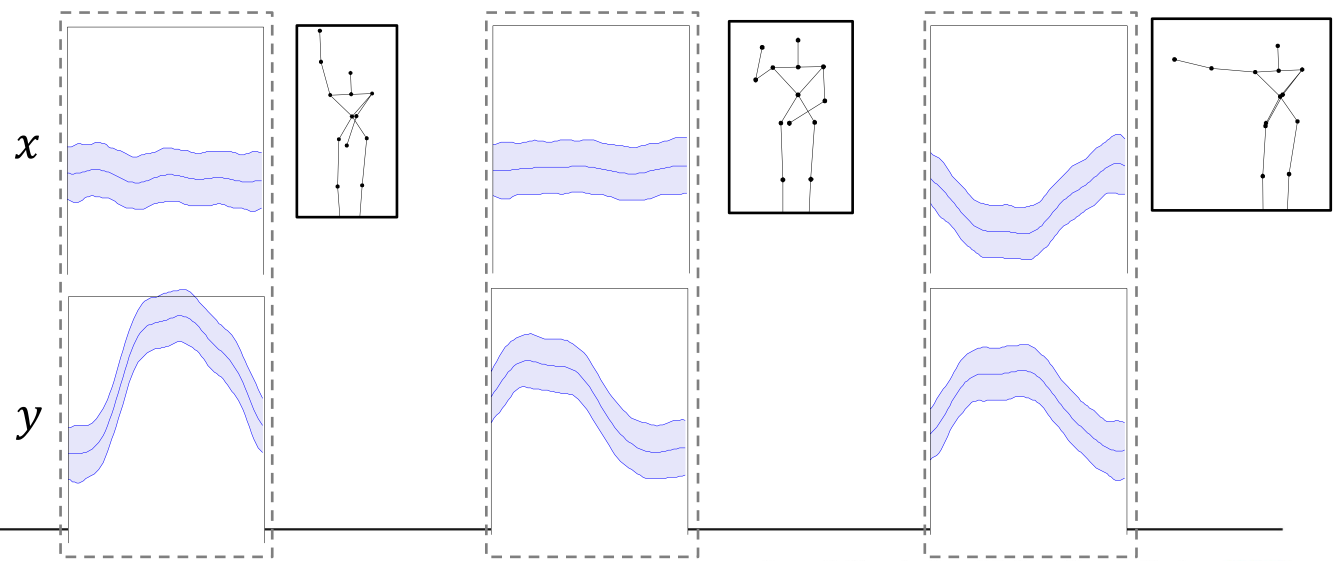


Results

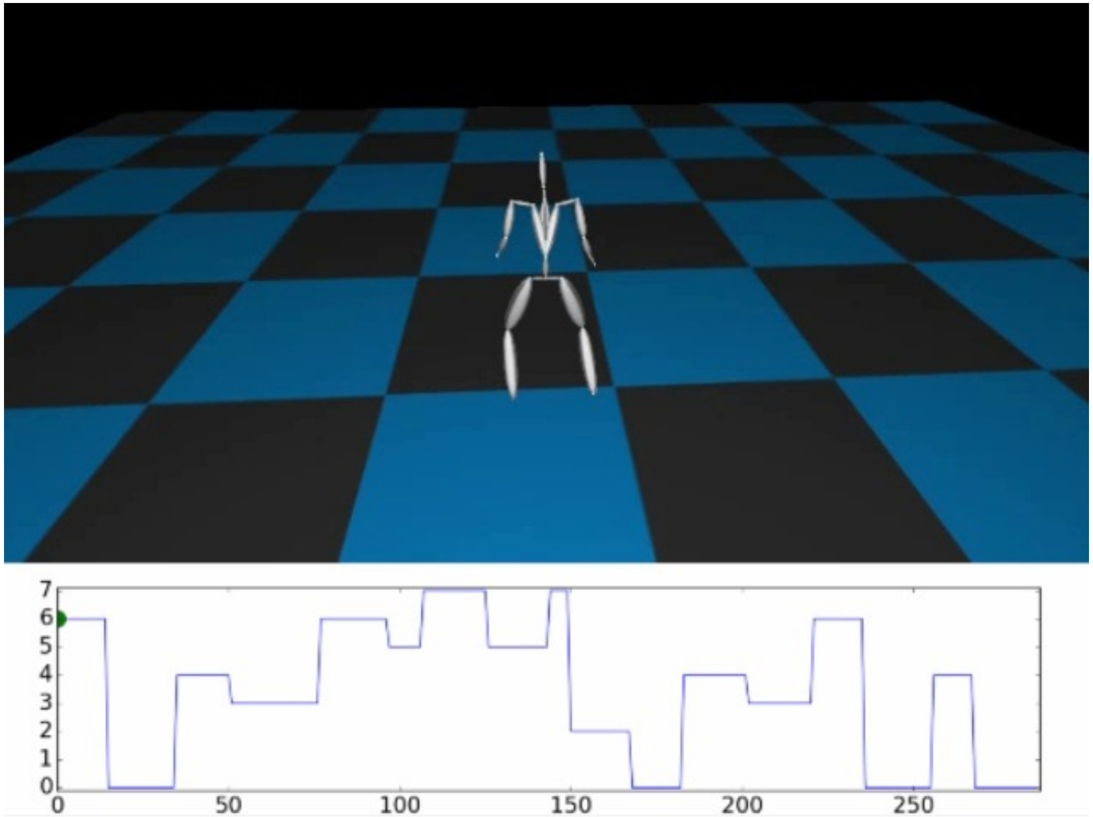
- Correctly recovered the three motions



- Learned Gaussian processes:



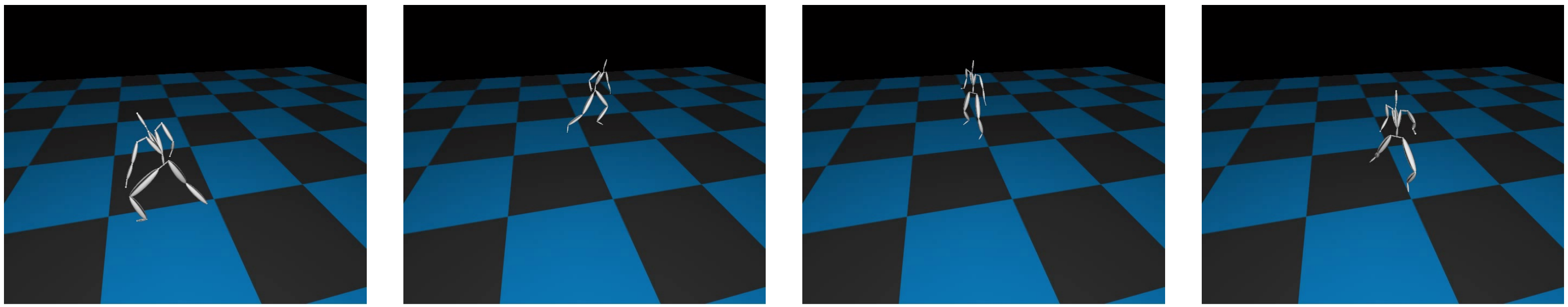
Motion segmentation



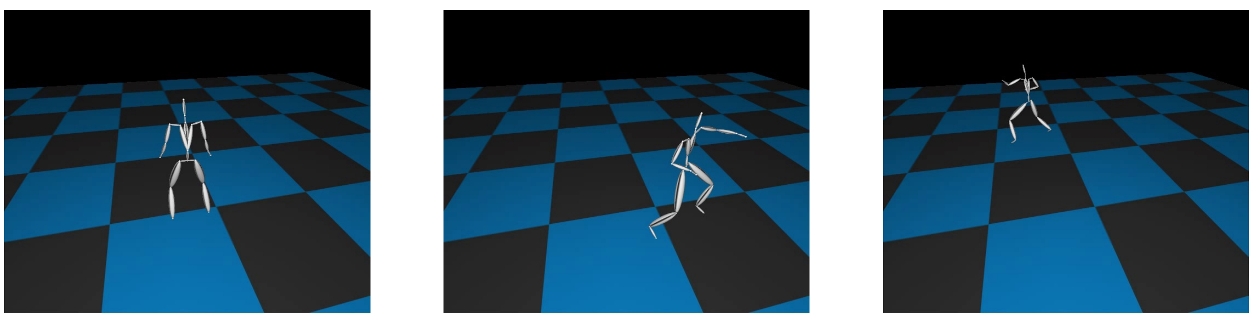
- Correctly recognized Karate motions from observations

Inferred motions (excerpt)

- Class 0: Right punch with an additional step

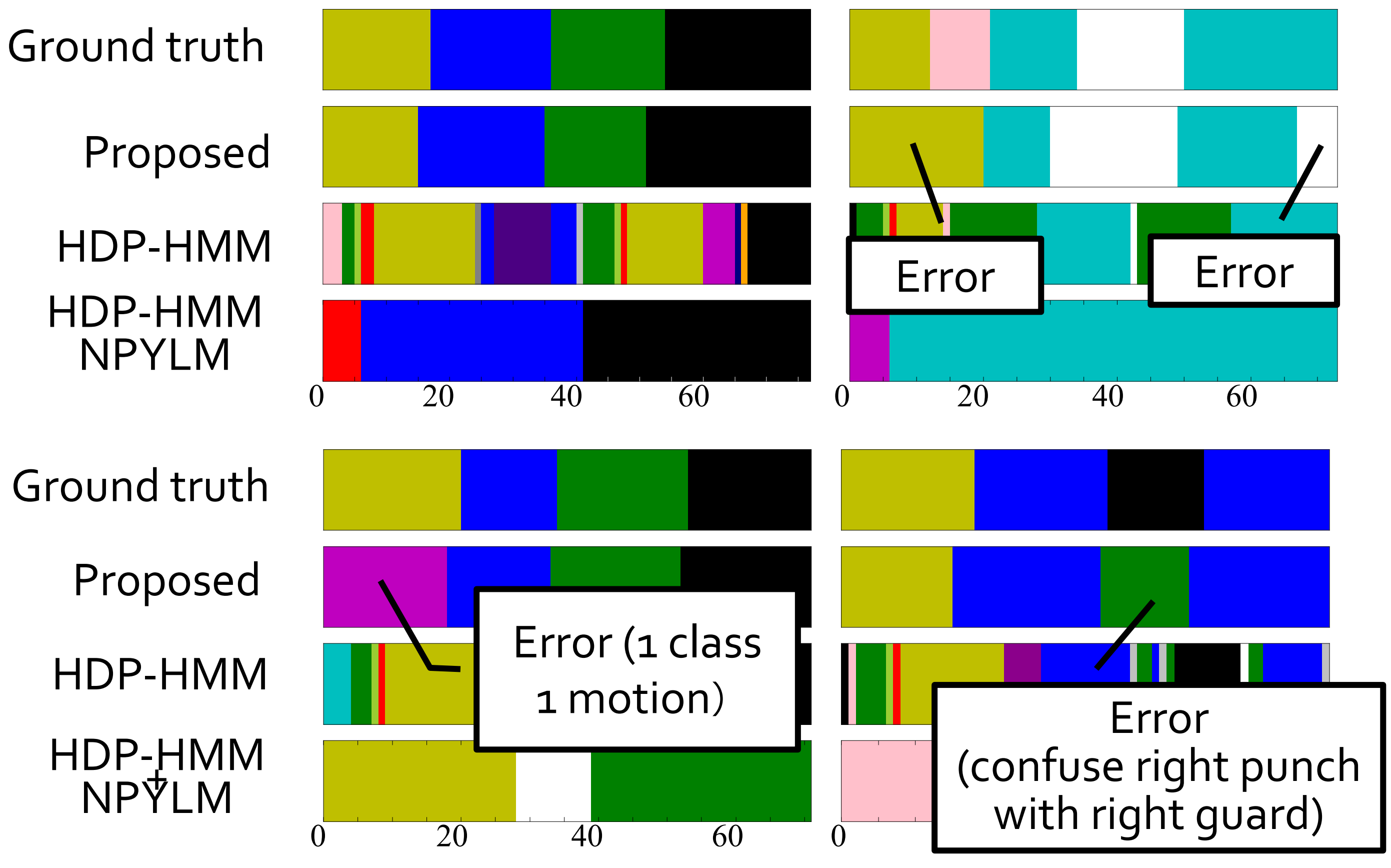


- Class 6: Left lower guarding



Comparison with other methods

- GP-HSMM vs. HDP-HMM, HDP-HMM+NPYLM



Number of Motions

- Number of hidden states (motions) can also be estimated using hierarchical Dirichlet processes
  - HDP-GP-HSMM
  - "Sequence Pattern Extraction by Segmenting Time Series Data Using GP-HSMM with Hierarchical Dirichlet Process", Nagano+, IROS 2018

TABLE V

SEGMENTATION RESULTS FOR THE EXERCISE MOTION.

	Hamming distance	Precision	Recall	F-measure	# of estimated classes
HDP-GP-HSMM	0.31	0.38	0.95	0.55	10
HDP-HMM	0.82	0.070	1.0	0.13	14
HDP-HMM+NPYLM	0.63	0.61	1.0	0.76	26
BP-HMM	0.23	0.25	1.0	0.40	18
Autoplait	0.61	0.67	0.18	0.28	5

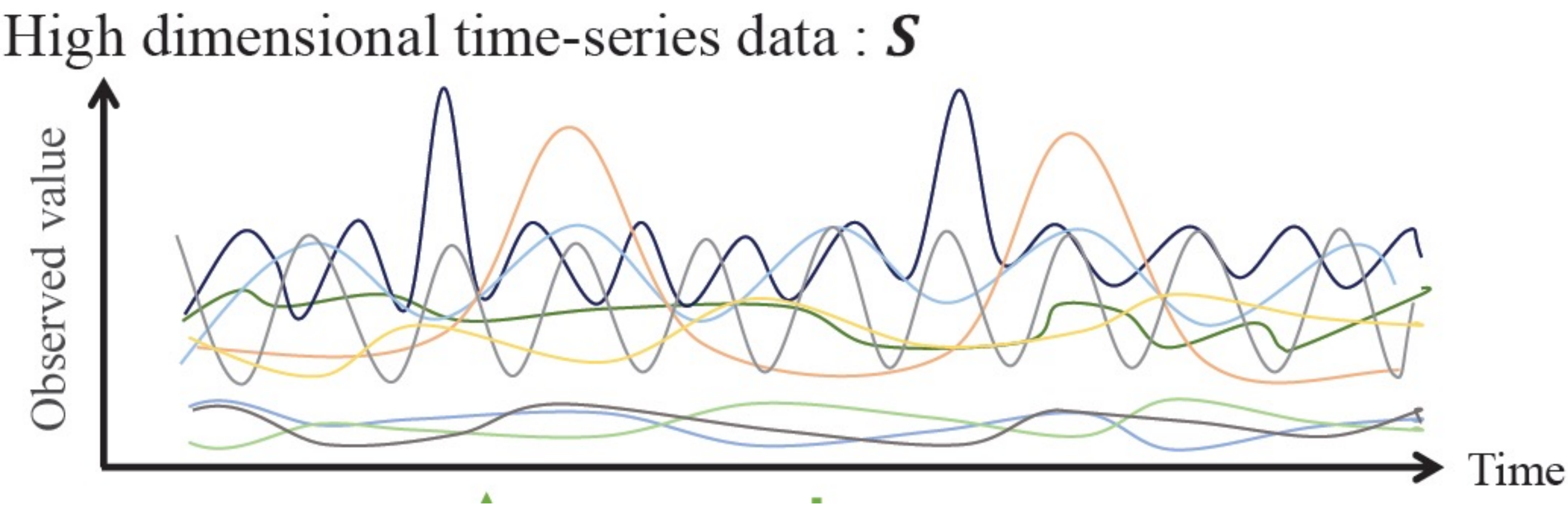
Ground truth: 11

Number of Motions (2)

- We leveraged infinite HMM (Beal+ 2001, Teh+ 2006) with our semi-Markov structure
- Since the state space is infinite-dimensional, also used a Beam sampling (van Gael+2007) for slice sampling+dynamic programming
  - Internally,using a stick-breaking representation for possibly infinite state spaces

High-dimensional regime

- Actually, robot movements are quite high-dimensional
  - In our case, # of joint angles = 93
  - Cannot apply the method to whole data



Strategy (Our work at IROS 2019)

- Solution: dimensionality reduction
- Linear PCA → NG
- GPLVM (Gaussian process LVM) → OK, but inference is not stable

- Using VAE as a surrogate of GPLVM

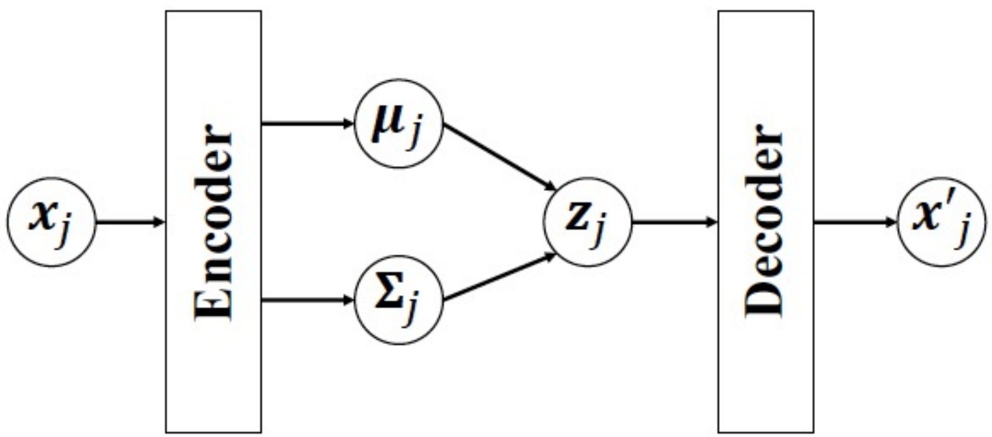
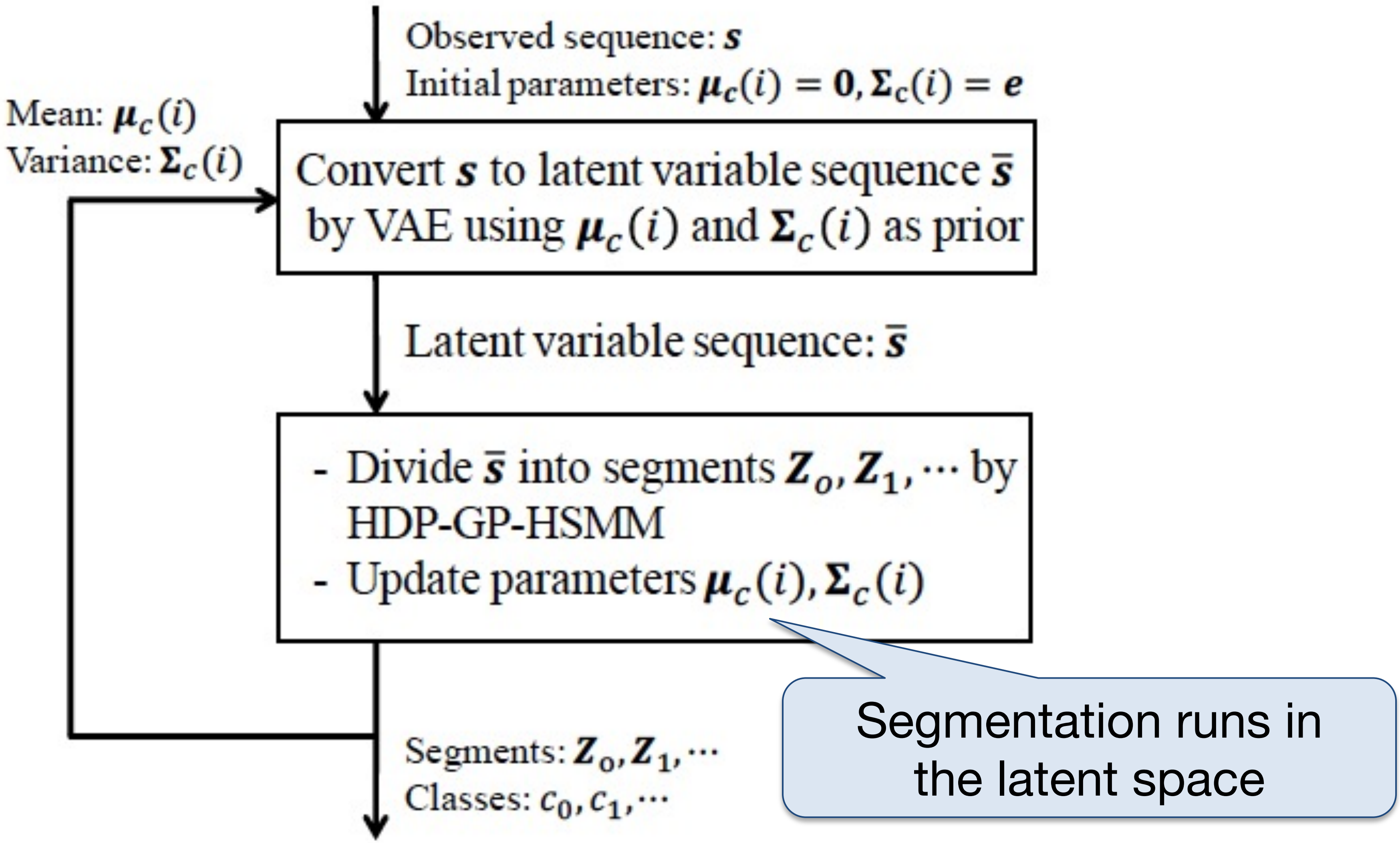


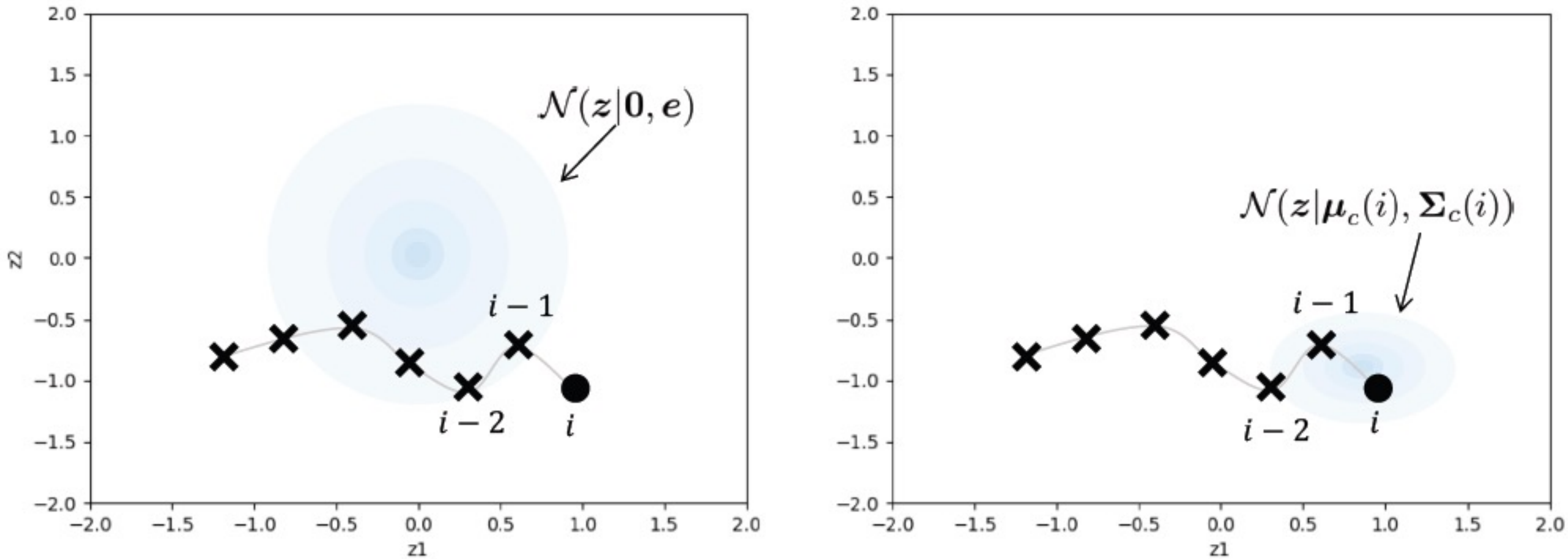
Fig. 3. Variational autoencoder (VAE) to obtain the latent low-dimensional representation z_j of observed time series x_j.

Simultaneous optimization



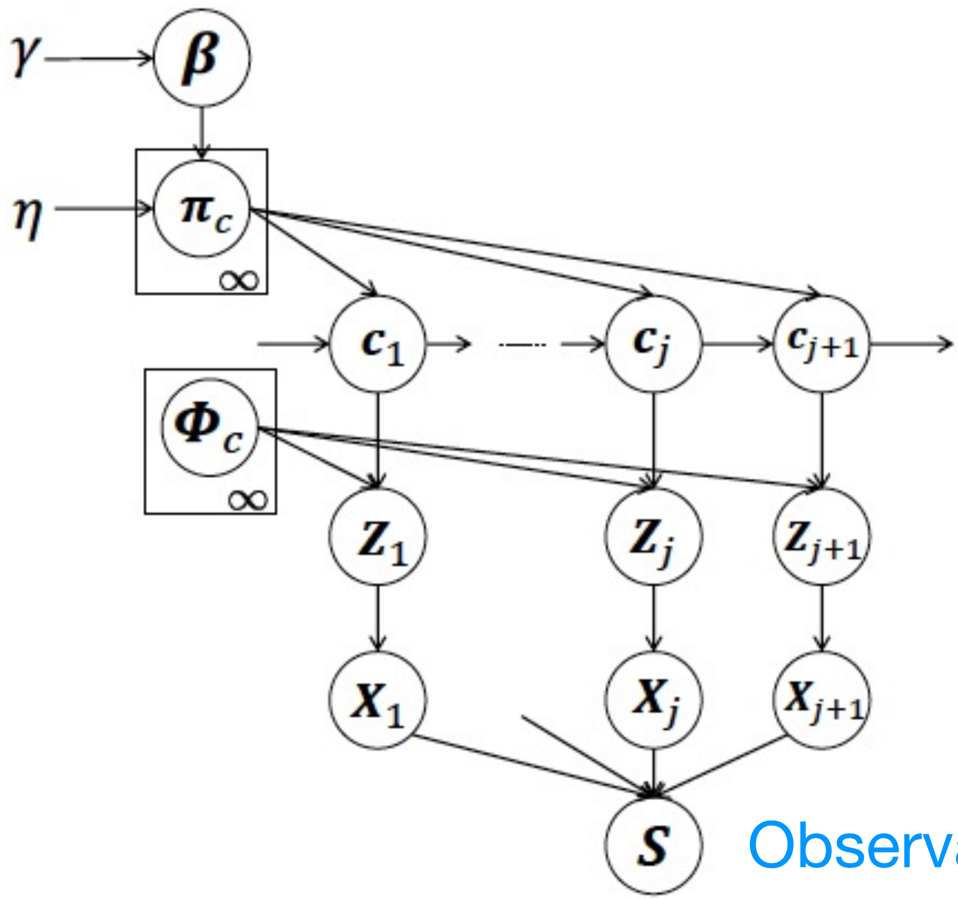
Note

- In this case, each cluster (=motion) has its own VAE for compression
  - VAE priors are different for each cluster



Graphical model

- We observe only ***s*** (high-dimensional time series)



Observation

Hierarchical Dirichlet process-variational autoencoder-Gaussian process-hidden semi-Markov model (Long!)
=
HVGH

Experiments

- Dance exercises which include four and seven unit motions (labels are not used in learning)

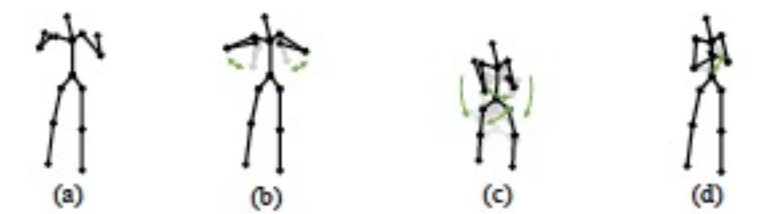


図 4: Four unit motions included in the chicken dance: (a) beaks, (b) wings, (c) tail feathers, and (d) claps.

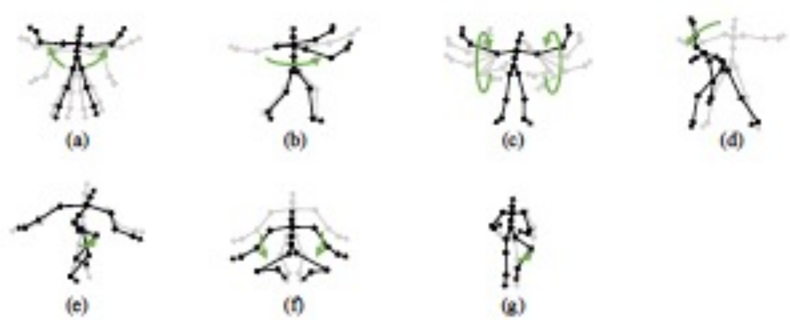


図 5: Seven unit motions included in the exercise motion1: (a) jumping jack, (b) twist, (c) arm circle, (d) bend over, (e) knee raise, (f) squatting, and (g) jogging

Results (1)

- Exercise containing four unit motions:

表 1: Segmentation results for the chicken dance.

	Hamming distance	Precision	Recall	F-measure	# of estimated classes
HVGH	0.23	0.86	0.86	0.86	4
VAE+HDP-GP-HSMM	0.31	1.0	0.71	0.83	4
VAE+HDP-HMM	0.74	0.15	1.0	0.26	11
VAE+ HDP-HMM+NPYLM	0.48	1.0	0.86	0.92	7
VAE+BP-HMM	0.34	1.0	0.86	0.92	3
VAE+Autoplait	0.66	0.0	0.0	0.0	1

VAE as a preprocessing

Results (2)

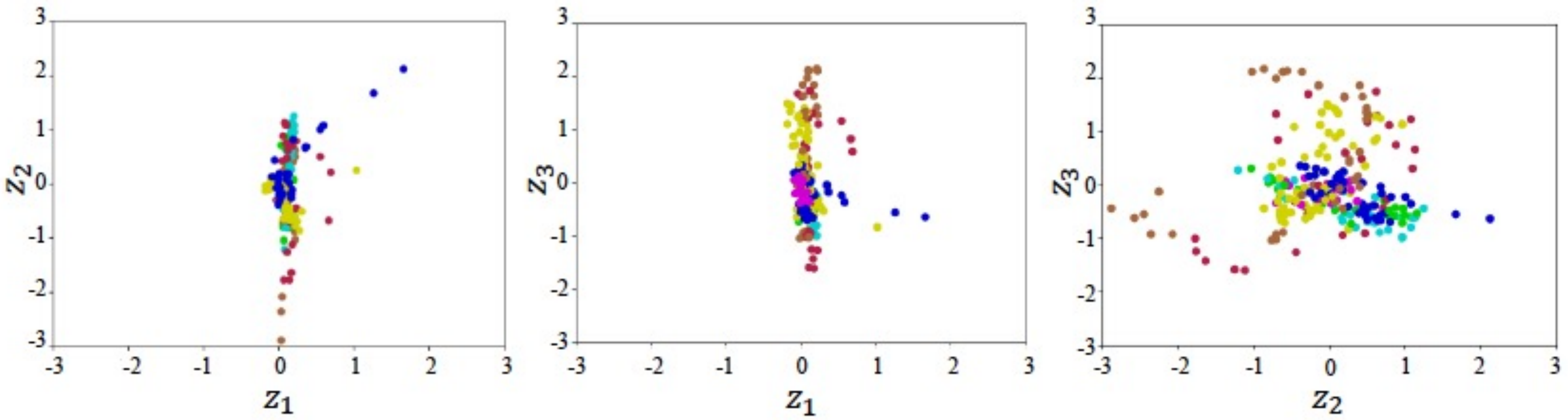
- Exercise containing seven unit motions:

表 2: Segmentation results for the exercise motion.

	Hamming distance	Precision	Recall	F-measure	# of estimated classes
HVGH	0.16	0.66	0.93	0.75	11
VAE+HDP-GP-HSMM	0.24	0.53	0.93	0.67	12
VAE+HDP-HMM	0.75	0.05	1.0	0.09	10
VAE+ HDP-HMM+NPYLM	0.61	0.30	1.0	0.45	28
VAE+BP-HMM	0.58	0.29	0.97	0.44	7
VAE+Autoplait	0.76	0.0	0.0	0.0	2

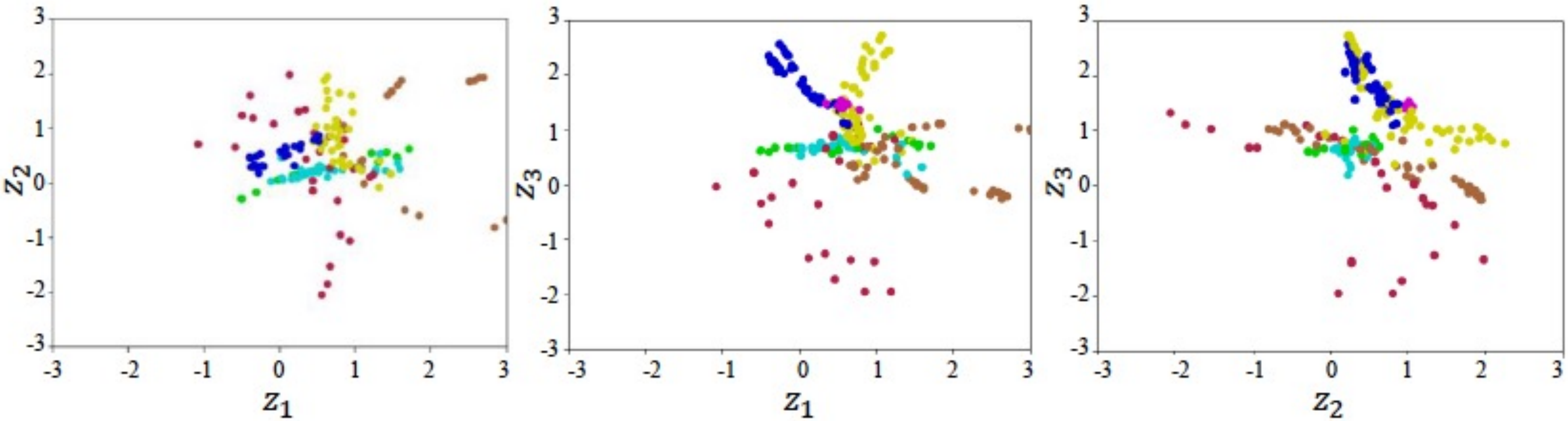
VAE as a preprocessing

Estimated latent space in VAE



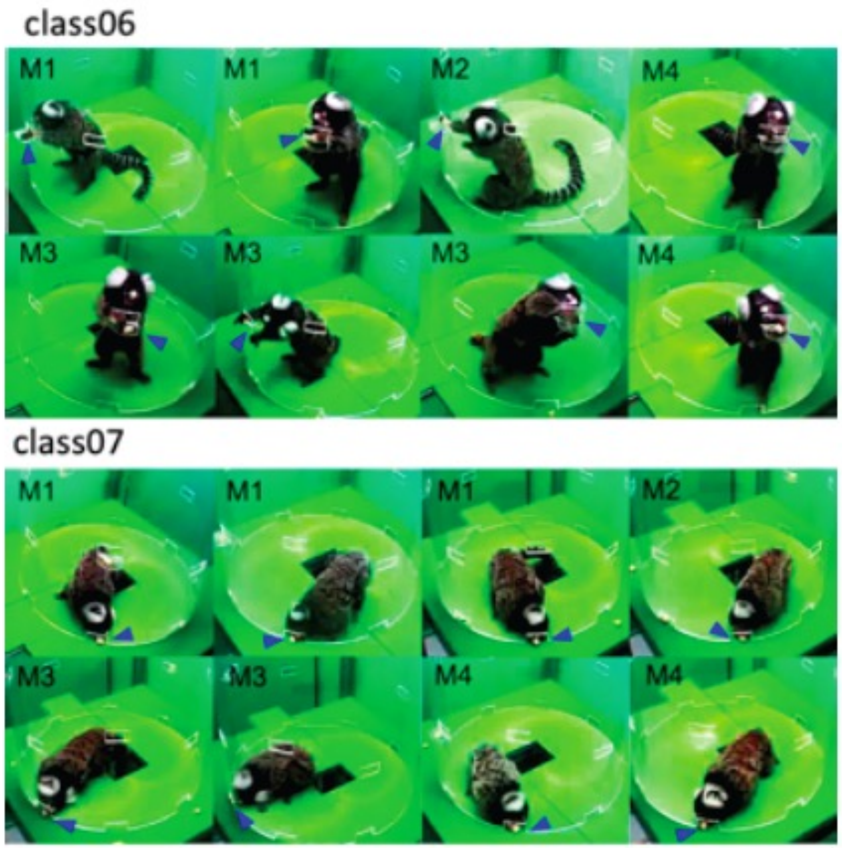
- VAE is separately learned for compression
- Motions (in color) are mixed and not separated

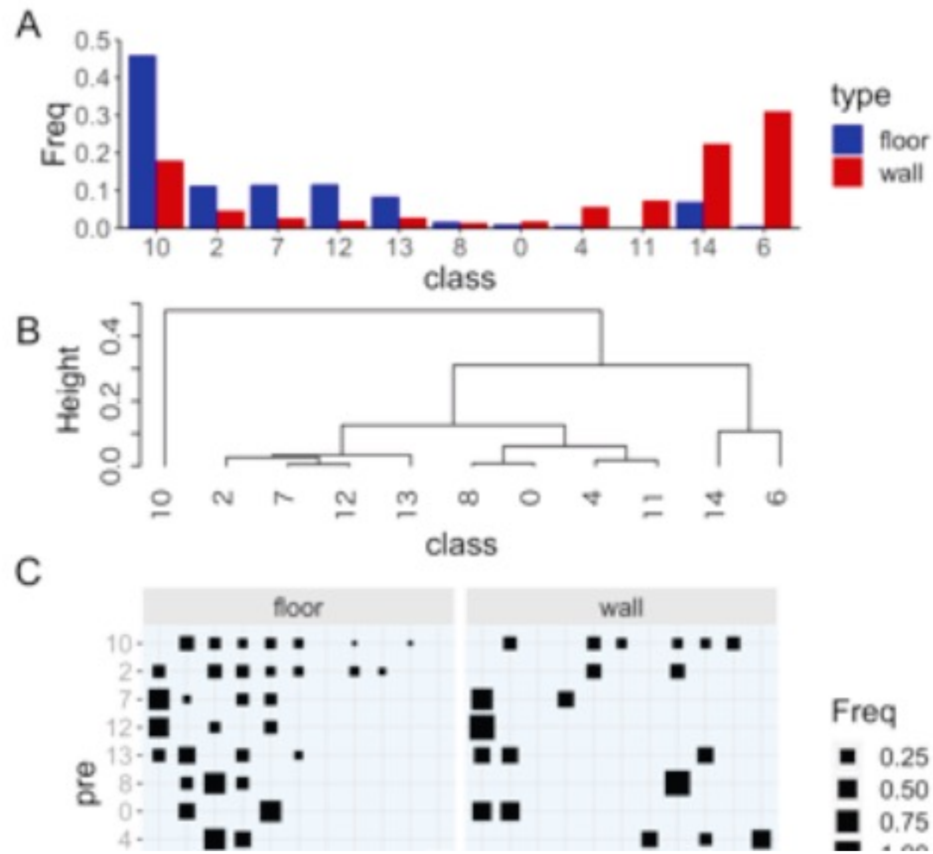
Estimated latent space in VAE (2)



- Simultaneous learning in HVGH
- Motions (in color) are clearly separated

Zoology and Brain science (Marmoset)





- Can recognize ape’s motions *automatically*
- joint work with Koki Mimura (NCAP), JSAI 2019

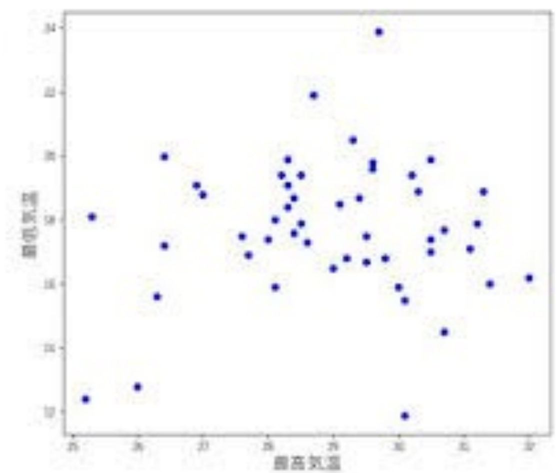
Part 3:
Nonparametric Bayesian Deep Visualization (NPDV)

Joint work with Bridgestone Corporation; appeared at IBISML 43 (2021)

Data Visualization

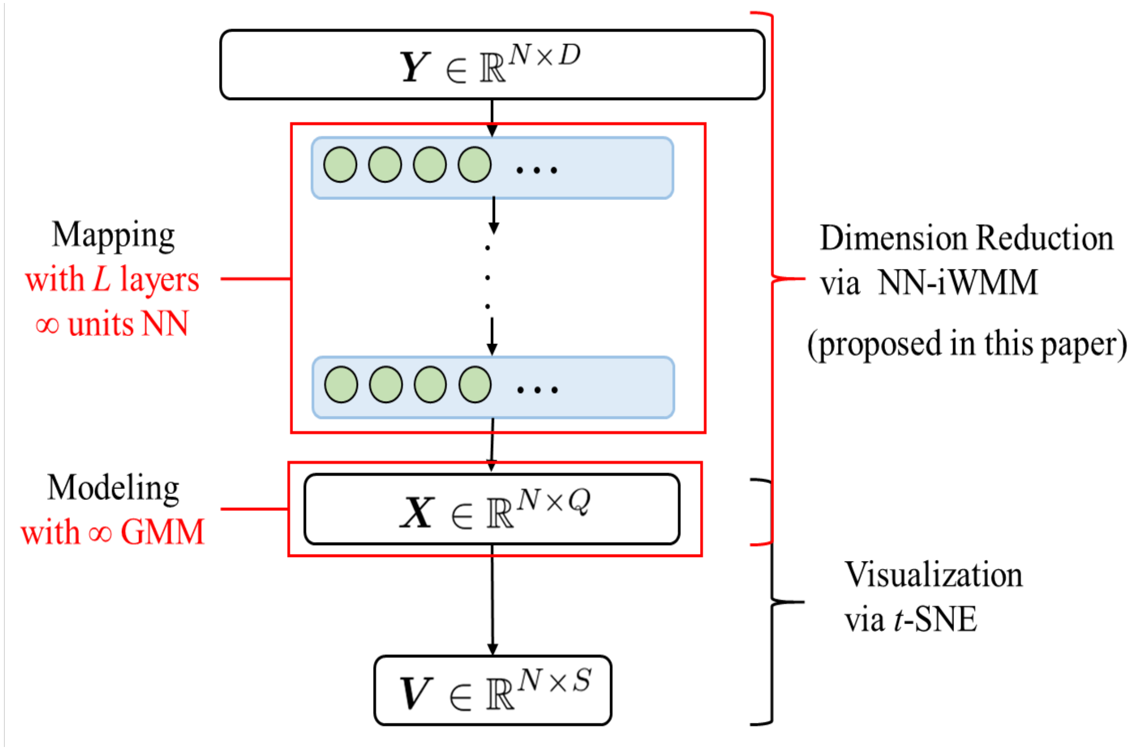
- Visualize high-dimensional data by mapping them into 2D or 3D
  - Important first step for exploratory data analysis

ID	Y1	Y2	…	Y_D
1				
2				
・・・				
N				



- Principal Component Analysis (PCA) has a strong limitation → Data often have nonlinear structures

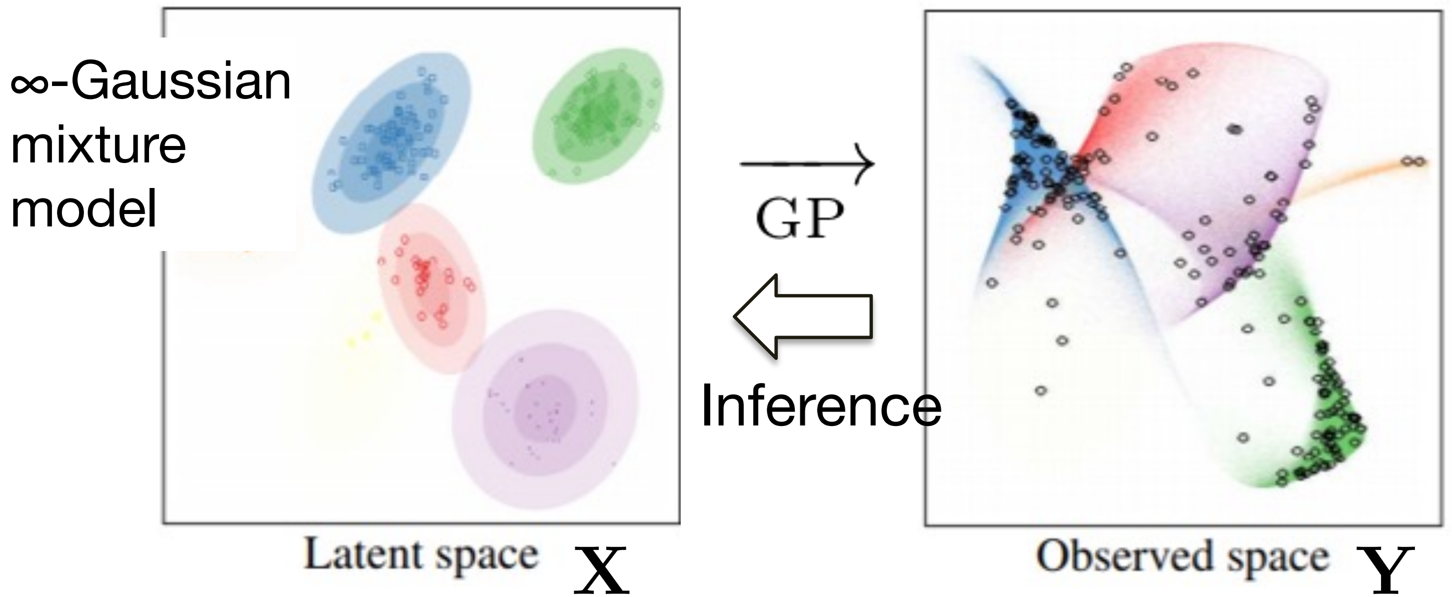
NPDV: NP Bayesian Deep Visualization



- Simultaneously optimize $\boldsymbol{Y} \to \boldsymbol{X},\ \boldsymbol{X} \to \boldsymbol{V}$
- $\boldsymbol{Y} \to \boldsymbol{X}$: NN-iWMM Neural network using GP + infinite mixture (no NN weights)
- $\boldsymbol{X} \to \boldsymbol{V}$: visualization via t-SNE (van der Maaten 2009)

Infinite Warped Mixture Model (iWMM)

(Iwata+ 2013)

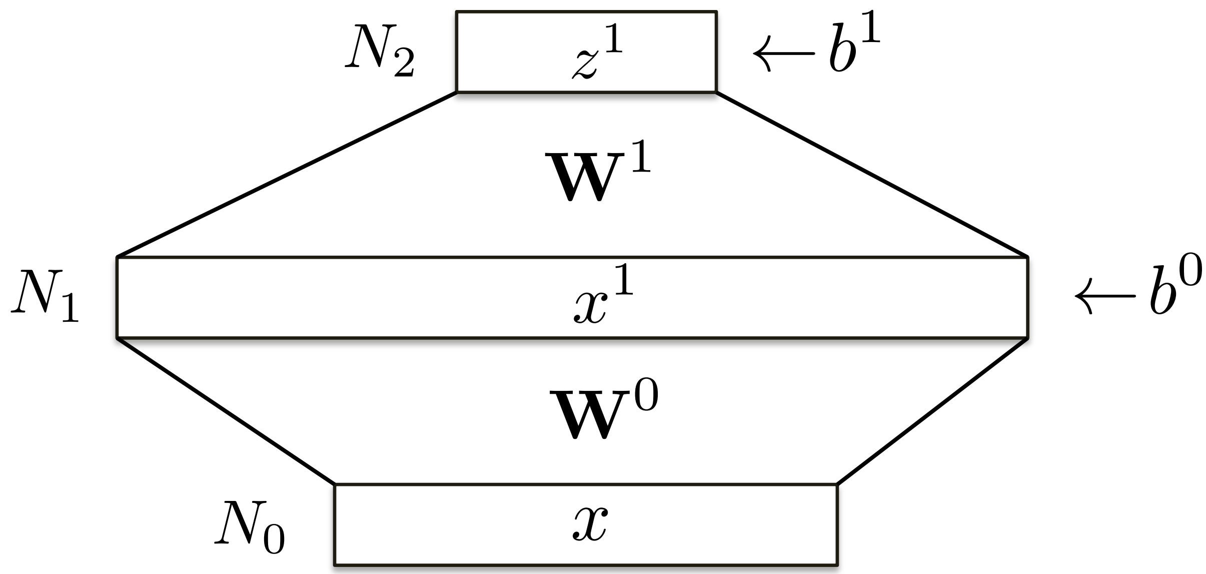


- Infinite Gaussian mixture in latent space
  → Map to observation through Gaussian process
- d'th dimension of observation *Y* distributes according to GP on X: $Y_d \sim \mathrm{GP}(\boldsymbol{\mu}, K_{\mathbf{X}})$

Neural network = Gaussian process

- Single hidden-layer NN with input layer x
- *i*'th output $z_i^1(x)$ is written as :

$$z_i^1(x) = b_i^1 + \sum_{j=1}^{N_1} W_{ij}^1 x_j^1(x), \quad x_j^1(x) = \phi\left(b_j^0 + \sum_{k=1}^{N_0} W_{jk}^0 x_k\right)$$



Connection

$$W_{ij}^\ell \sim \mathcal{N}(0, \sigma_w / N_\ell)$$

Bias

$$b_i^\ell \sim \mathcal{N}(0, \sigma_b)$$

Neural network = Gaussian process (2)

$$z_i^1(x) = b_i^1 + \sum_{j=1}^{N_1} W_{ij}^1 x_j^1(x), \quad x_j^1(x) = \phi\left(b_j^0 + \sum_{k=1}^{K} W_{jk}^0 x_k\right)$$

- Sum of independent weights W_{ij}^ℓ and biases b_i^ℓ
  → Joint probability $p(z_1^1(x), z_2^1(x), \cdots, z_{N_2}^1(x))$
  has a multivariate Gaussian distribution
  by Central Limit Theorem $\cdots$ Gaussian process
  – Mean: clearly 0
  – Covariance is

$$K^1(x, x') \equiv \mathbb{E}\left[z_i^1(x) z_i^1(x')\right]$$

$$= \sigma_b^2 + \sigma_w^2 \mathbb{E}\left[x_i^1(x) x_i^1(x')\right] = \sigma_b^2 + \sigma_w^2 C(x, x')$$

NNGP (Neural Network Gaussian process)

- Assume $\ell - 1$ layer output $z_j^{\ell-1}$ is GP:

$$z_i^\ell(x) = b_i^\ell + \sum_{j=1}^{N_\ell} W_{ij}^\ell \, x_j^\ell(x), \quad x_j^\ell(x) = \phi(z_j^{\ell-1}(x))$$

- ℓ -layer Mean is 0, variance is

$$\begin{aligned} K^\ell(x, x') &\equiv \mathbb{E}\left[z_i^\ell(x) z_i^\ell(x')\right] \\ &= \sigma_b^2 + \sigma_w^2 \, \mathbb{E}_{z_i^{\ell-1} \sim \mathrm{GP}(0, K^{\ell-1})} \left[\phi(z_i^{\ell-1}(x)) \phi(z_i^{\ell-1}(x'))\right] \end{aligned}$$

  – This expectation can be computed by:
    (1) GP regression (2) Numerical approximation
    (3) Analytical solution for specific ϕ like ReLU

NNGP (Neural Network Gaussian process)

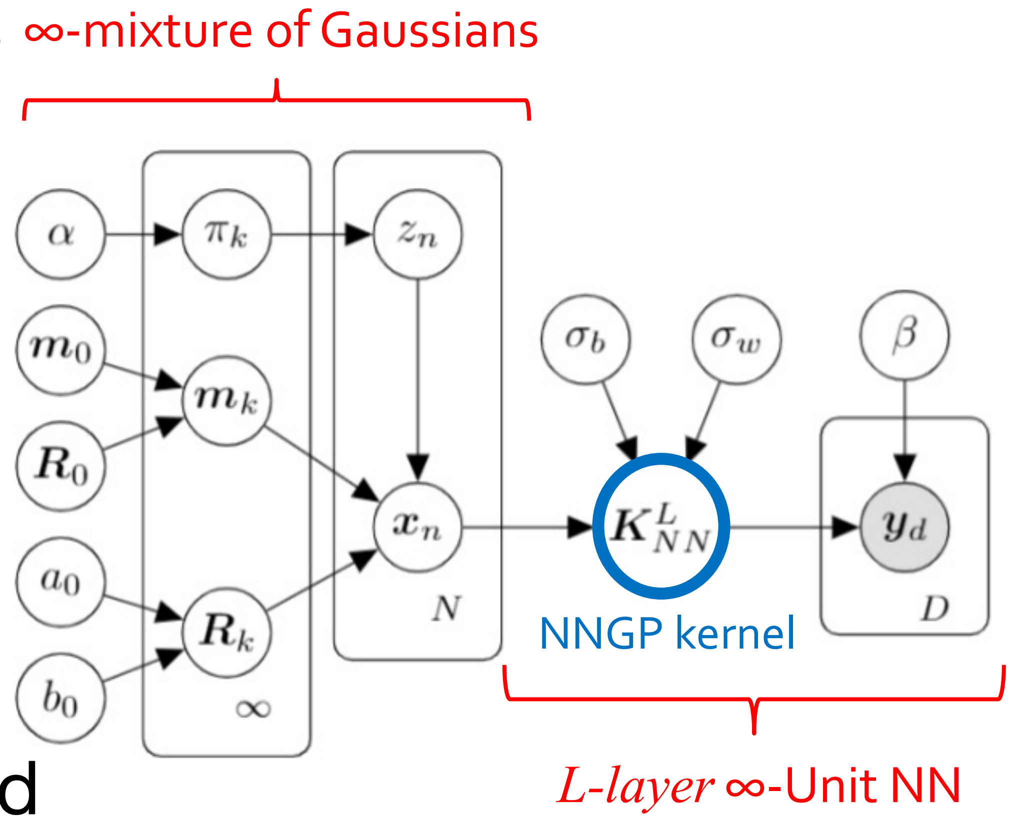
- When ϕ is ReLU (Cho&Saul 2009, Lee+ 2017) :

$$
\begin{aligned}
K^{\ell}(x,x') = \sigma_b^2 + \frac{\sigma_w^2}{2\pi}\sqrt{K^{\ell-1}(x,x)K^{\ell-1}(x',x')} \\
\times \left(\sin\theta_{x,x'}^{\ell-1} + (\pi - \theta_{x,x'}^{\ell-1})\cos\theta_{x,x'}^{\ell-1}\right)
\end{aligned}
$$

$$
\theta_{x,x'}^{\ell} = \cos^{-1}\left(\frac{K^{\ell}(x,x')}{\sqrt{K^{\ell}(x,x)K^{\ell}(x',x')}}\right)
$$

Generative model of NN-iWMM

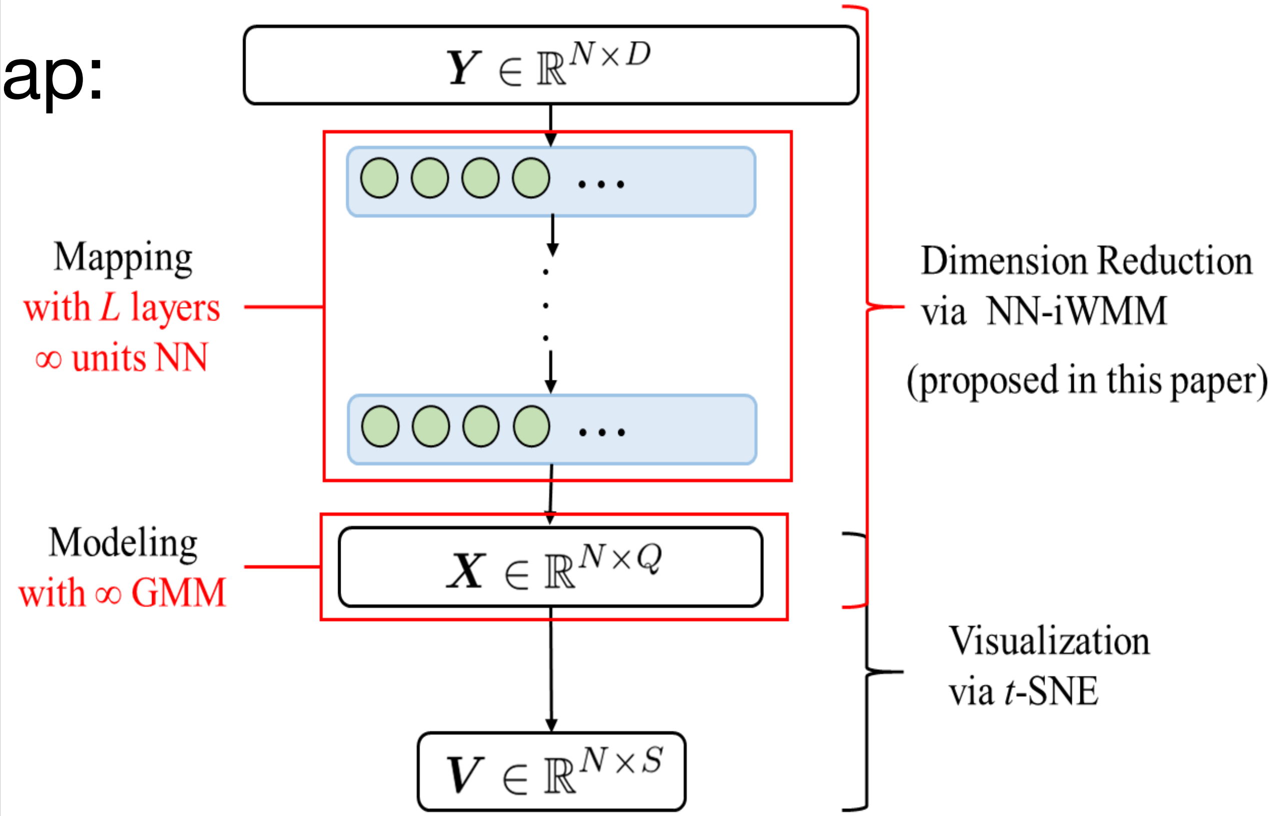
- ∞-Gaussian mixture generates latents
  $$\boldsymbol{X} \equiv \{\boldsymbol{x}_i\}_{i=1}^N$$
- From $\boldsymbol{X}$ and kernel parameters σ_b, σ_w , compute NNGP kernel matrix K_{NN}^L
- Generate data $\boldsymbol{Y}$ for each dimension d from NNGP



Unsupervised NN with no units,
no numerical weights

Joint modeling with *t*-SNE

- Recap:



- Problem: *t*-SNE has no clear generative model
  → How to combine NN-iWMM and *t*-SNE?

Joint modeling with t-SNE: RegBayes

- Posterior $p(\boldsymbol{\theta}|\boldsymbol{Y})$ equals to following optimization: (A. Zellner, 1988)

$$\begin{cases} \min\limits_{q(\boldsymbol{\theta})} & \mathrm{KL}[q(\boldsymbol{\theta})\|p(\boldsymbol{\theta})] - \int q(\boldsymbol{\theta}) \log p(\boldsymbol{Y}|\boldsymbol{\theta}) d\boldsymbol{\theta} \\ \text{s.t.} & q(\boldsymbol{\theta}) \in \mathcal{P} \end{cases}$$

$\boldsymbol{\theta}$: parameters
$\mathcal{P}$: probability distribution

- RegBayes：Bayesian model with posterior constraint (J. Zhu+. 2014)

$$\begin{cases} \min\limits_{q(\boldsymbol{\theta})} & \mathrm{KL}[q(\boldsymbol{\theta})\|p(\boldsymbol{\theta})] - \int q(\boldsymbol{\theta}) \log p(\boldsymbol{Y}|\boldsymbol{\theta}) d\boldsymbol{\theta} \\ \text{s.t.} & E_{q(\boldsymbol{\theta})}[\mathcal{R}(\boldsymbol{\theta}, \boldsymbol{Y})] \leq 0,\ q(\boldsymbol{\theta}) \in \mathcal{P} \end{cases}$$

$\mathcal{R}(\boldsymbol{\theta}, \boldsymbol{Y})$: Regularization term

  - Optimal distribution under the given constraint

$$q^*(\boldsymbol{\theta}) \propto \boxed{p(\boldsymbol{Y}|\boldsymbol{\theta})p(\boldsymbol{\theta}) \exp(-\lambda \mathcal{R}(\boldsymbol{\theta}, \boldsymbol{Y}))}$$

Joint distribution of $\boldsymbol{\theta}$ and $\boldsymbol{Y}$

NPDV Formulation

- Joint distribution of NPDV ($q^*(\theta)$ and NN-iWMM, t-SNE)

$$q^*(\boldsymbol{\theta}) \propto \underbrace{p(\boldsymbol{Y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}_{\text{NN-iWMM}} \exp(-\lambda \underbrace{\mathcal{R}(\boldsymbol{\theta}, \boldsymbol{Y})}_{t\text{-SNE}})$$

- Specifically,

$$\left.\begin{aligned}
&p(\boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{z}, \{\boldsymbol{m}_k, \boldsymbol{R}_k \pi_k\}_{k=1}^{\infty} | \boldsymbol{V}) \\
&\propto p(\boldsymbol{Y}|\boldsymbol{X}) p(\boldsymbol{X}|\boldsymbol{z}, \{\boldsymbol{m}_k, \boldsymbol{r}_k, \pi_k\}_{k=1}^{\infty}) p(\boldsymbol{z}|\{\pi_k\}_{k=1}^{\infty}) \\
&\times p(\{\boldsymbol{m}_k\}_{k=1}^{\infty}) p(\{\boldsymbol{r}_k\}_{k=1}^{\infty}) p(\{\pi_k\}_{k=1}^{\infty})
\end{aligned}\right\} \text{NN-iWMM}$$

$$\times \exp(-\lambda \mathrm{KL}[\boldsymbol{p}^X \| \boldsymbol{p}^V]) \quad t\text{-SNE between } \boldsymbol{X} \text{ and } \boldsymbol{V}$$

  - $\boldsymbol{V}$: visual coordinates, $\sigma_w, \ \sigma_b$: NNGP kernel hyperparameter
  - λ : hyperparameter ※ *λ=ND in this study*

NPDV training

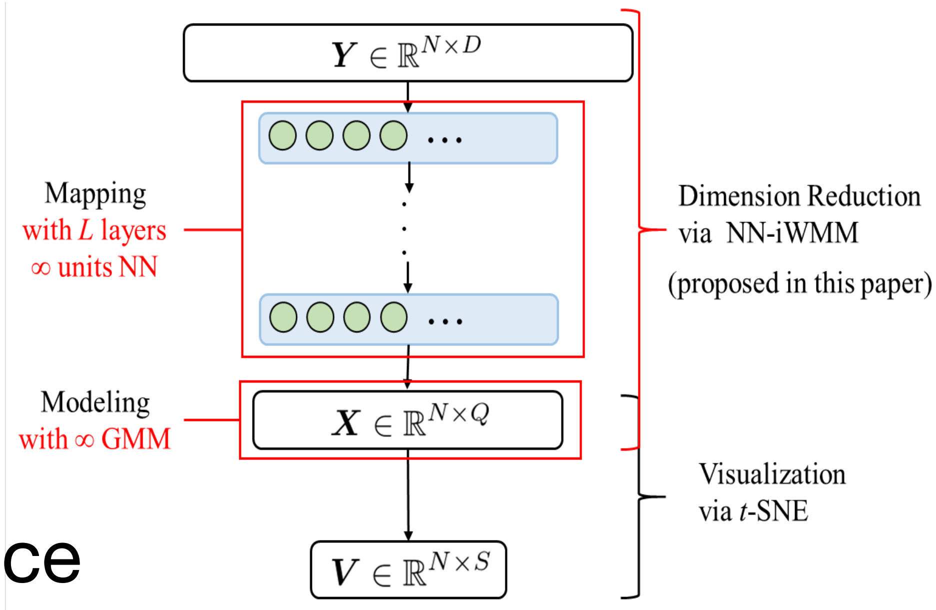
- Objective : lower bound of likelihood using variational distribution Q

$$
\begin{aligned}
\mathcal{L} &= \mathbb{E}_{q(\boldsymbol{X})}[\log p(\boldsymbol{Y}|\boldsymbol{X})] - \mathbb{E}_{q(\boldsymbol{X})}[\log q(\boldsymbol{X})] \\
&+ \mathbb{E}_{q(\boldsymbol{X},\boldsymbol{z},\boldsymbol{m},\boldsymbol{\Sigma},\boldsymbol{\phi})}\left[\log \frac{p(\boldsymbol{X},\boldsymbol{z},\{\boldsymbol{m}_k,\boldsymbol{r}_k,\pi_k\}_{k=1}^{K})}{q(\boldsymbol{z},\{\boldsymbol{m}_k,\boldsymbol{R}_k\}_k,\boldsymbol{\pi})}\right] - \lambda\mathbb{E}_{q(\boldsymbol{X})}[\mathrm{KL}[\boldsymbol{p}^X\|\boldsymbol{p}^V]] \\
&= \underbrace{\mathcal{L}_1 + \mathcal{L}_2 - \lambda\mathcal{R}}_{(*)} + \underbrace{\mathcal{H}(q(\boldsymbol{X}))}_{\text{Gauss Entropy}}
\end{aligned}
$$

  - Re-parametrization trick to generate randomly from $q(\boldsymbol{x}_n) = \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_n, \boldsymbol{S}_n)$ to approximate (*)
  - Gradient base learning from the approximation

GP Neural Networks and ARD

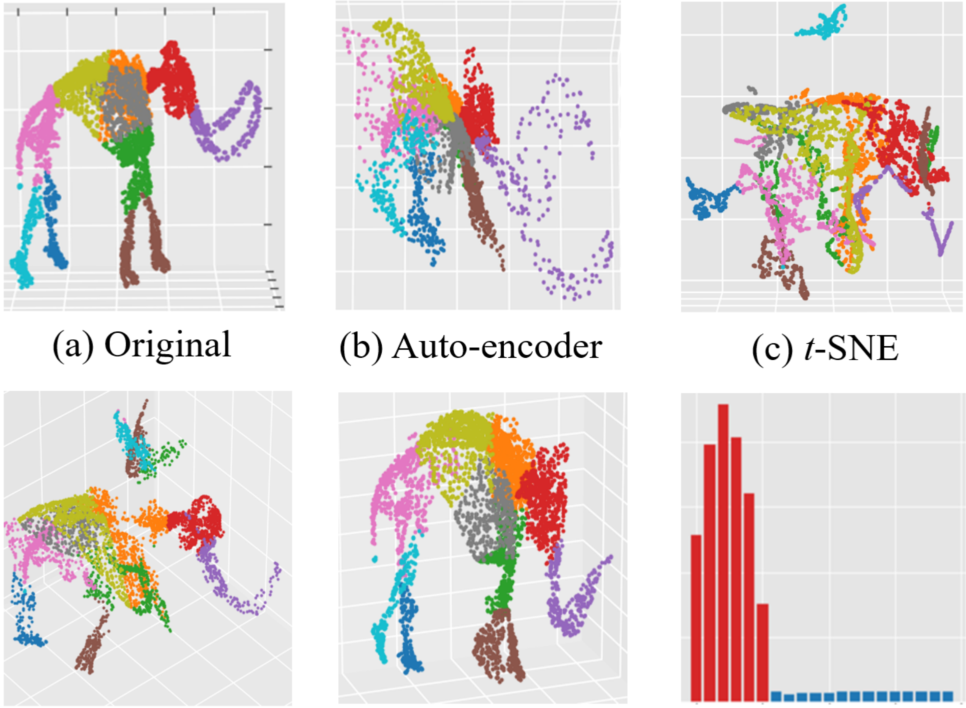
- Since there is no NN weights in NPDV, we can directly optimize hidden X for observation Y



- ARD (automatic relevance determination) prior on X
  → We can learn hidden dimensionality of X!
  – This is impossible for ordinary neural networks with numerous weights to learn

GP Neural Networks and ARD

- Input: “mammoth” 3D data mapped non-linearly for high dimensions
  - NPDV correctly recovered mammoth, with effectively three dimensions!



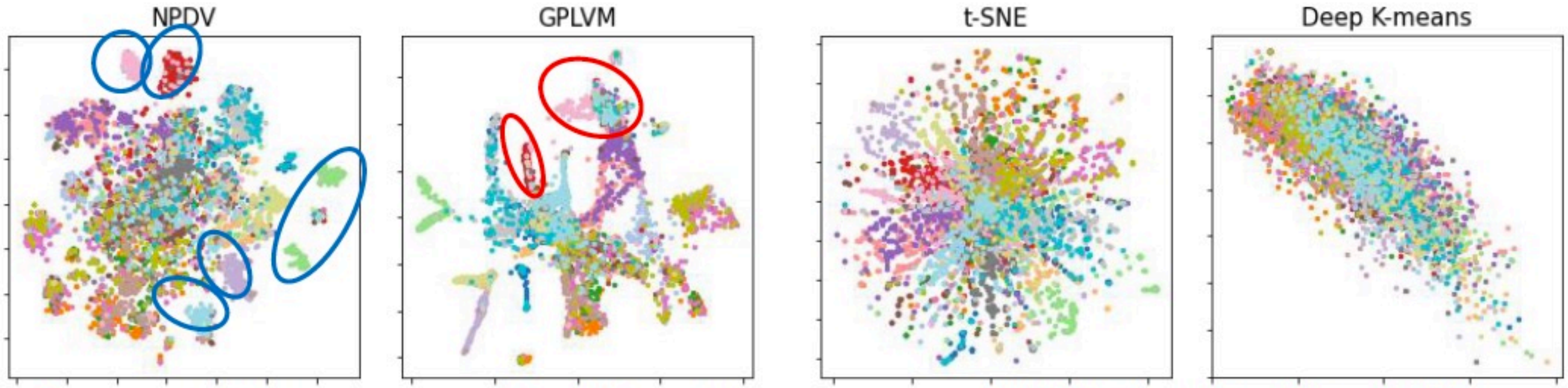
(a) Original (b) Auto-encoder (c) *t*-SNE

(d) NN-iWMM (e) NN-iWMM+PCA (f) ARD weights

Experiments

- Text corpus: 20newsgroups (English)、Livedoor News corpus (Japanese)
  - 20newsgroups: USENET internet news text on 20 newsgroups, 18000 documents
  - Livedoor: Livedoor news articles, 9 categories, 7300 documents
- tf.idf preprocessing, compress into 1000 dims to visualize in 2D
- Baseline: t-SNE (Maaten+ 2009), Deep K-means (Phard+ 2020)

Results (20newsgroups: English)



○：5つの明確なクラスタを確認

×：赤、桃クラスタと他のクラスタが重複

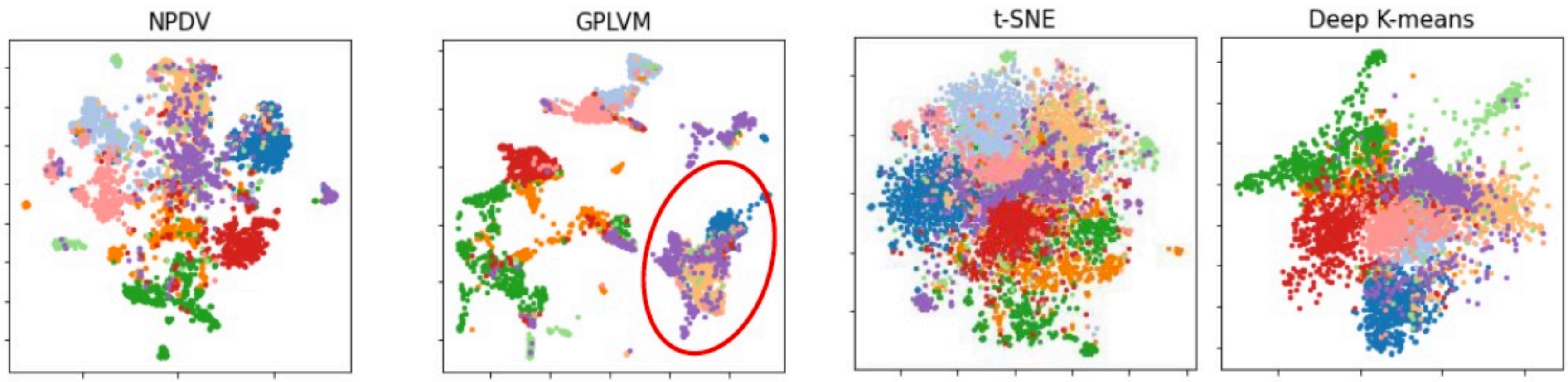
×：全体的にクラスターが互いに重複（特にDeep k-means）

- *k*-NN classification accuracy (Bold: SOTA, ＊：Next to SOTA)

近傍数	Deep *k*-means	GPLVM	t-SNE	*t*-SNE-GPLVM	提案手法（次点との差分）
k = 10	0.383	0.527	0.559	0.593 *	**0.599** (＋0.6%)
k = 20	0.347	0.496	0.512	0.529 *	**0.552** (+ 2.3%)
k = 30	0.330	0.487	0.492	0.500 *	**0.529** (+ 2.9%)

Clearer idendification of clusters & SOTA accuracy

Results (Livedoor: Japanese)



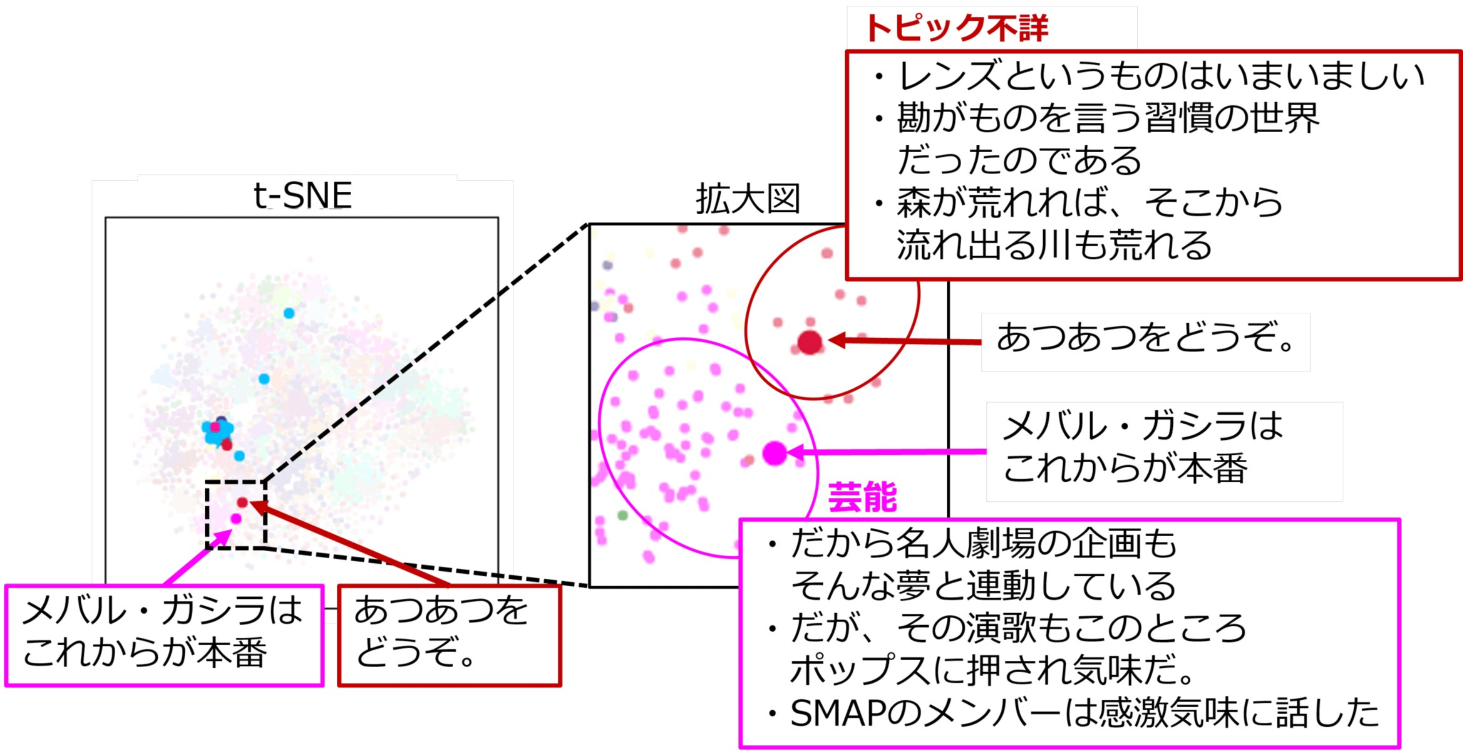
○ : クラスタの混合が少ない　× : 青、紫、薄橙が混合　× : クラスターが互いにオーバーラップ

- *k*-NN classification accuracy (Bold: SOTA, ＊ : Next to SOTA)

近傍数	Deep *k*-means	*t*-SNE-GPLVM	t-SNE	GPLVM	提案手法（次点との差分）
k = 10	0.755	0.793	0.793	0.809 *	**0.844** (+ 3.5%)
k = 20	0.746	0.767	0.767	0.798 *	**0.822** (+ 2.4%)
k = 30	0.742	0.747	0.747	0.790 *	**0.809** (+ 1.9%)

Unsupervised visualization of sentences (*t*-SNE)

- Directly visualize sentences via *t*-SNE
  → Prone to noise, cannot consider meanings



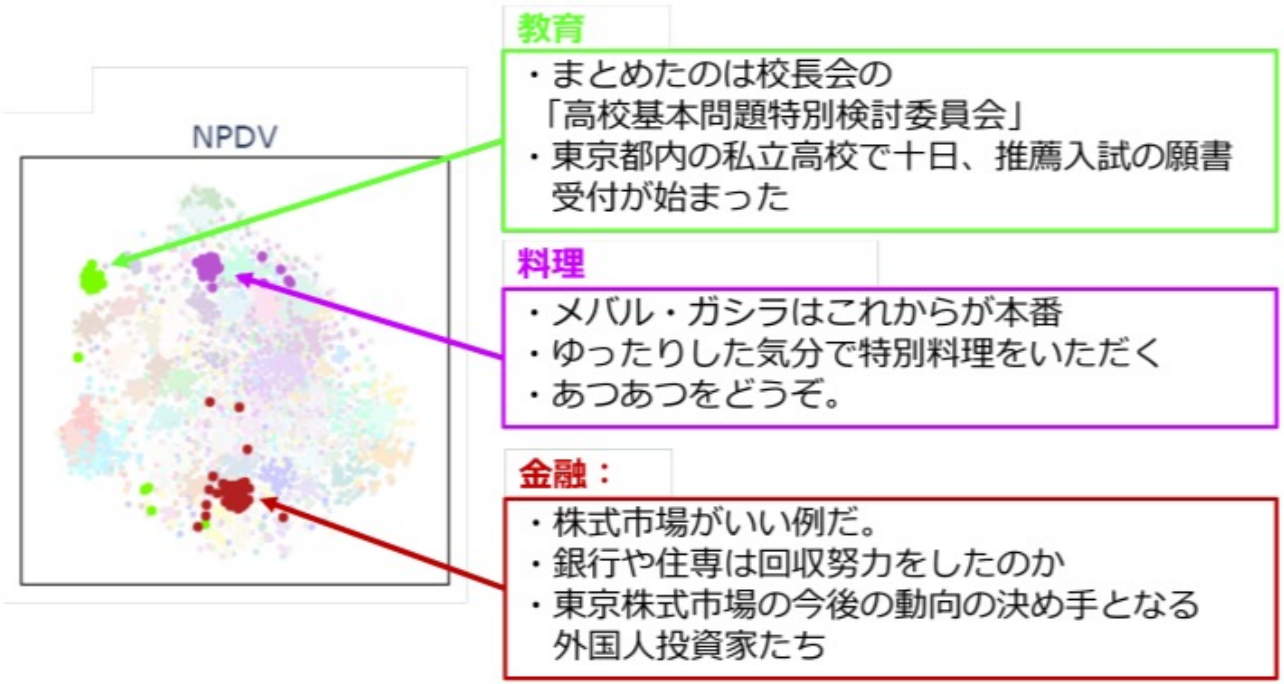
× Unrelated clustering of sentences that belong to “cooking” cluster under NPDV

Unsupervised visualization of sentences (NPDV)

- Internally estimates ∞-GMM to give “label” on each sentence

Clusters obtained through NPDV

<u>Simultaneously estimates latent clustering and visualization</u>



Data:
Kyoto corpus
(5000 articles from Mainichi newspaper 1995)

○ NPDV estimates semantically meaningful clusters

Summary

- Gentle introduction to Gaussian processes
- Learning “Motions” from movements through a Gaussian process generative model for robots (from research on NLP)
  - GP+HDP+Hidden semi-Markov Model (IROS 2018)
  - HVGH: GP+HDP+Hidden SMM+VAE (IROS 2019) for high-dimensional observations (segmentation on latent Gaussian processes)
- Analytical Gaussian process neural network with no neural weights
  - NPDV: Visualization w/ combination with *t*-SNE

“Language and Robotics”

- *Advanced Robotics* (2019),
  “Survey on frontiers of language and robotics”.
  T. Taniguchi, D. Mochihashi, T. Nagai, et al.
  https://www.tandfonline.com/doi/full/10.1080/01691864.2019.1632223