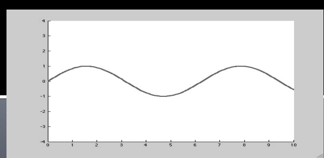
CSE-473: Gaussian Processes for Bayesian Filtering

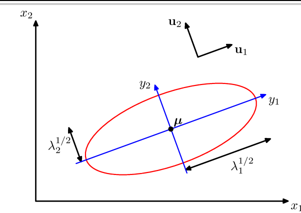


High-level Idea of GPs

- Non-parametric regression model
- Distribution over functions
- Fully specified by training data and kernel function
- Output variables are jointly Gaussian
- Covariance given by distance of inputs in kernel space

Picture from [Bishop: Pattern Recognition and Machine Learning, 2006]

Gaussians

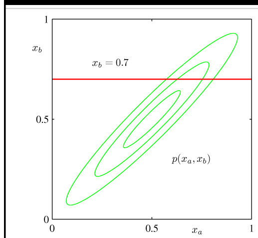


$p(\mathbf{x}) = \mathrm{N}(\mu, \Sigma)$

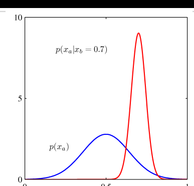
$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mu)^T \Sigma^{-1}(\mathbf{x}-\mu)}$$

Pictures from [Bishop: PRML, 2006]

Marginalization / Conditioning



$p(x_a) = \mathrm{N}(\mu_a, \Sigma_{aa})$



$p(x_a | x_b) = \mathrm{N}(\mu_{a|b}, \Sigma_{a|b})$

$\mu_{a|b} = \mu_a + \Sigma_{ab}\Sigma_{bb}^{-1}(x_b - \mu_b)$

$\Sigma_{a|b} = \Sigma_{aa} - \Sigma_{ab}\Sigma_{bb}^{-1}\Sigma_{ba}$

GP Setting

- Outputs are noisy function of inputs: $y_i = f(\mathbf{x}_i) + \varepsilon$
- GP prior: Outputs jointly zero-mean Gaussian:

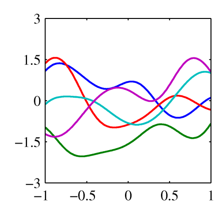
$p(\mathbf{y} | \mathbf{X}) = \mathrm{N}(\mathbf{0}, \mathbf{K} + \sigma_n^2 \mathbf{I})$

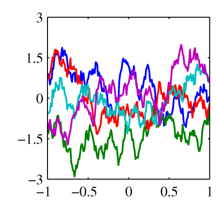
- Covariance given by kernel matrix over inputs:

$$\mathbf{K} = \begin{pmatrix} k(x_1,x_1) & \dots & k(x_1,x_n) \\ k(x_2,x_1) & & \\ \vdots & k(x_i,x_j) & \vdots \\ k(x_n,x_1) & \dots & k(x_n,x_n) \end{pmatrix} \qquad k(x,x') = \sigma_f^2 \, e^{-\frac{1}{2}(x-x')W(x-x')^T}$$

Pictures from [Bishop: PRML, 2006]

Functions Sampled from Prior





GP Prediction

- Training data:

$$D=\{(\mathbf{x}_1,y_1),(\mathbf{x}_2,y_2),\ldots,(\mathbf{x}_n,y_n)\}=(\mathbf{X},\mathbf{y})$$

- Prediction given training samples:

$$p(y^* \mid \mathbf{x}^*,\mathbf{y},\mathbf{X}) = N\left(\mu,\sigma^2\right)$$

$$\mu = \mathbf{k}^{*T}\left(\mathbf{K}+\sigma_n^2\mathbf{I}\right)^{-1}\mathbf{y}$$

$$\mathbf{k}^*[i] = k(\mathbf{x}^*,\mathbf{x}_i)$$

$$\sigma^2 = k(\mathbf{x}^*,\mathbf{x}^*) - \mathbf{k}^{*T}\left(\mathbf{K}+\sigma_n^2\mathbf{I}\right)^{-1}\mathbf{k}^*$$

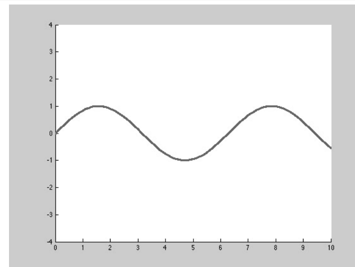
$$p(x_a \mid x_b) = \mathrm{N}\left(\mu_{a|b},\Sigma_{a|b}\right)$$

$$\mu_{a|b} = \mu_a + \Sigma_{ab}\Sigma_{bb}^{-1}(x_b-\mu_b)$$

$$\Sigma_{a|b} = \Sigma_{aa} - \Sigma_{ab}\Sigma_{bb}^{-1}\Sigma_{ba}$$

Recall conditional

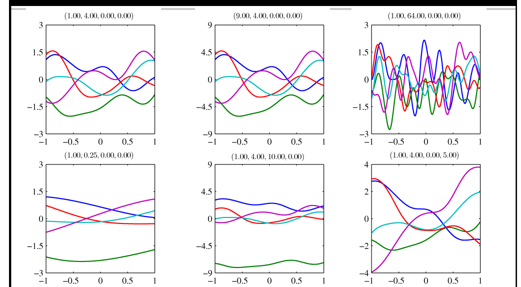
GP Prediction



Hyperparameters

Pictures from [Bishop: PRML, 2006]

$$k(x,x')=\sigma_f^2\, e^{-\frac{1}{2}(x-x')W(x-x')^T}$$



Hyperparameter Estimation

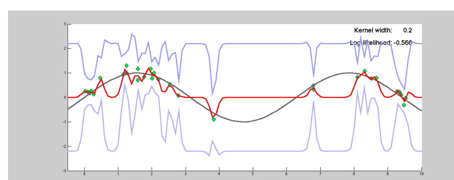
- Maximize data log likelihood:

$$\theta_* = \arg\max_\theta p(\mathbf{y} \mid \mathbf{X},\theta)$$

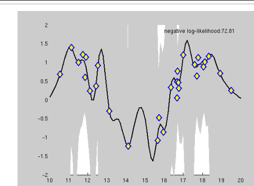
$$\log p(\mathbf{y} \mid \mathbf{X},\theta) = -\frac{1}{2}\mathbf{y}^T\left(\mathbf{K}+\sigma_n^2\mathbf{I}\right)^{-1}\mathbf{y} - \frac{1}{2}\log\left(\mathbf{K}+\sigma_n^2\mathbf{I}\right) - \frac{n}{2}\log 2\pi$$

- Compute derivatives wrt. params $\theta = \langle\sigma_n^2, l, \sigma_f^2\rangle$
- Optimize using conjugate gradient descent

Kernel Width



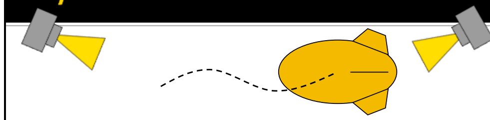
GP Optimization



- Learn hyperparameters via numerical methods
- Learn noise model at the same time

Bayesian Filtering with GPs

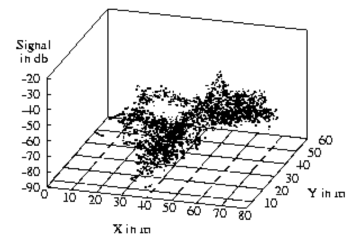
Dynamics and Observation Models



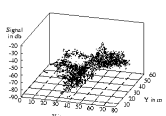
- System modeling
  - Learn behavior of the system given ground truth states
- Dynamics model
  - Discrete-time model of change over time
- Observation model
  - Mapping between states and observations
- Traditionally use parametric models
  - Derive system equations then learn parameters

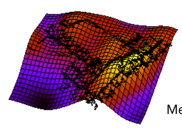
[Ferris-Haehnel-Fox: RSS-06]

WiFi Sensor Model

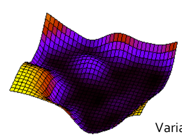


WiFi Sensor Model



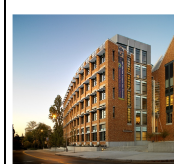


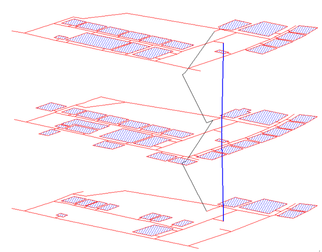
Mean



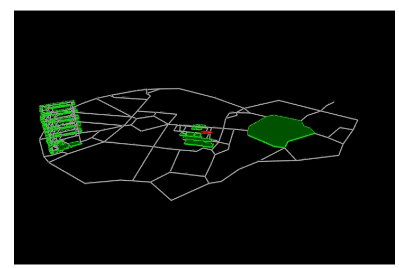
Variance

Mixed Representation

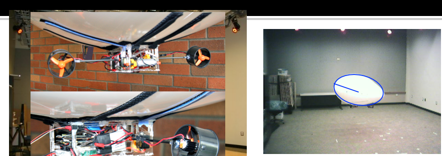




Tracking Example



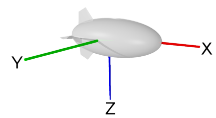
Blimp Test Platform



observation

- System:
  - Commercial blimp envelope with custom gondola
  - XScale based computer with Bluetooth connectivity
  - Two main motors with tail motor (3D control)
- Observations:
  - Two cameras each operating at 1Hz
  - Extract ellipse using computer vision (5D observations)
- Ground truth obtained via VICON motion capture system

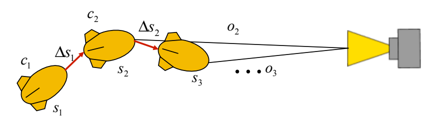
Non-linear Parametric Model



$$\dot{s} = \frac{d}{dt}\begin{bmatrix} p \\ \xi \\ v \\ \omega \end{bmatrix} = \begin{bmatrix} R_b^e v \\ H(\xi) \\ M^{-1}(\sum Forces - \omega * Mv) \\ J^{-1}(\sum Torques - \omega * J\omega) \end{bmatrix}$$

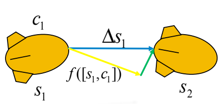
- 12-D state=[pos,rot,transvel,rotvel]
- Describes evolution of state as ODE
- Forces / torques considered: buoyancy, gravity, drag, thrust
- 16 parameters are learned by optimization on ground truth motion capture data

Learning GP Dynamics and Observation Models



- Use ground truth state to extract:
  - Dynamics data
    $D_S = \langle [s_1, c_1], \Delta s_1 \rangle, \langle [s_2, c_2], \Delta s_2 \rangle \ \dots$
  - Observation data
    $D_O = \langle s_2, o_2 \rangle, \langle s_3, o_3 \rangle$
- Learn models using Gaüssian process regression
  - Learn process noise inherent in system

Learning Enhanced-GP Models



- Combine GP model with parametric model
  $D_X = \langle [s_1, c_1], \Delta s_1 - f([s_1, c_1]) \rangle$
- Advantages
  - Captures aspects of system not considered by parametric model
  - Learns noise model in same way as GP-only models
  - Higher accuracy for same amount of training data

GP Modeling Accuracy

Dynamic model error

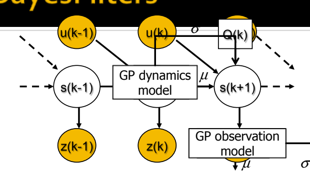
Propagation method	pos(mm)	rot(deg)	vel(mm/s)	rotvel(deg/s)
Param	3.3	0.5	14.6	1.5
GPonly	1.8	**0.2**	9.8	**1.1**
EGP	**1.6**	**0.2**	**9.6**	1.3

Observation model error

Modeling method	pos(pix)	Major axis(pix)	Minor axis(pix)	Theta(deg)
Param	7.1	2.9	5.7	9.2
GPonly	4.7	3.2	**1.9**	**9.1**
EGP	**3.9**	**2.4**	**1.9**	9.4

- 1800 training points, mean error over 900 test points
- For dynamic model, 0.25 sec predictions

GP-BayesFilters

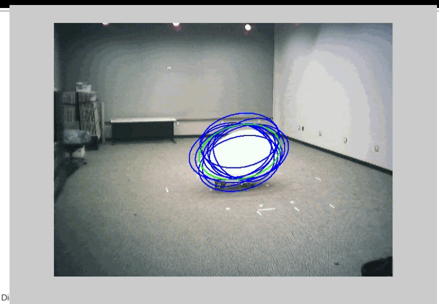


- Traditional Bayesian filtering
  - Parametric dynamics and observation models
- GP-BayesFilters
  - GP dynamics and observation models
  - Noise derived from GP prediction uncertainty
  - Can be integrated into to Bayes filters: EKF, UKF, PF, ADF

GP-BayesFilters

- Learn GP:
  - Input: Sequence of ground truth states along with controls and observations: <s, u, z>
  - Learn GPs for dynamics and observation models
- Filters
  - Particle filter: sample from dynamics GP, weigh by Gaussian GP observation function
  - EKF: GP for mean state, GP derivative for linearization
  - UKF: GP for sigma points

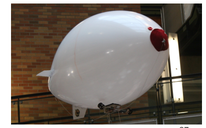
GP-UKF Tracking Example



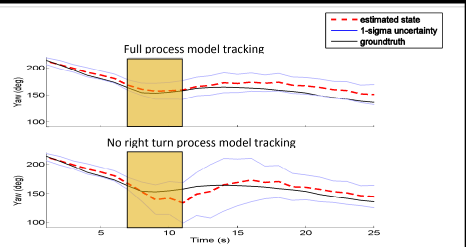
Blimp Results

Tracking algorithm	pos(mm)	rot(deg)	vel(mm/s)	rotvel(deg/s)	MLL	time(sec)
GP-PF	91+/-7	6.4+/-1.6	52+/-3.7	5.0+/-.2	9.4+/-1.9	449.4+/-21
GP-EKF	93+/-1	5.2+/-.1	52+/-.5	4.6+/-.1	13.0+/-.2	.29+/-.1
GP-UKF	89+/-1	4.7+/-.2	50+/-.4	4.5+/-.1	14.9+/-.5	1.28+/-.3
ParaPF	115+/-5	7.9+/-.1	64+/-1.2	7.6+/-.1	-4.5+/-4.2	30.7+/-5.8
ParaEKF	112+/-4	8.0+/-.2	65+/-2	7.5+/-.2	8.4+/-1	.21+/-.1
ParaUKF	111+/-4	7.9+/-.1	64+/-1	7.6+/-.1	10.1+/-1	.33+/-.1

- Blimp tracking using multiple cameras
- Ground truth obtained via Vicon motion tracking system
- Average tracking error
- Trajectory ~12 min long
- 0.5 sec timesteps



Dealing With Training Data Sparsity



- Training data for right turns removed

Going Latent

[Lawrence: NIPS-03,JMLR-05]

- Sometimes ground truth states are not or only partially available.
- Instead of optimizing over hyperparameters only
  $\theta_* = \arg\max_\theta p(\mathbf{y} \mid \mathbf{X}, \theta)$
  optimize over latent states *X* as well
  $\langle \mathbf{X}_*, \theta_* \rangle = \arg\max_{\mathbf{X},\theta} p(\mathbf{y} \mid \mathbf{X}, \theta)$
- GPLVM: non-linear probabilistic dimensionality reduction

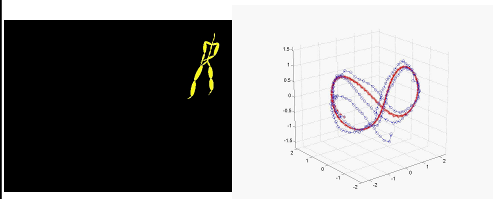
GPDM: Latent Variable Models for Dynamical Systems

[Wang-Fleet-Hertzmann: PAMI-08]

- GPLVM with additional constraints on latent variables to model dynamical system
  $p(\mathbf{y}, \mathbf{X}, \theta) = p(\mathbf{y} \mid \mathbf{X}, \theta)\, p(\mathbf{X} \mid \theta)\, p(\theta)$
  $\langle \mathbf{X}_*, \theta_* \rangle = \arg\max_{\mathbf{X},\theta} p(\mathbf{y}, \mathbf{X}, \theta)$
- Dynamics are modeled via another GP

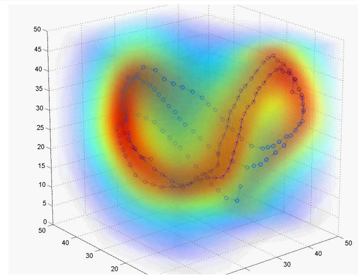
GPDM Application

[Wang-Fleet-Hertzmann: PAMI-08]



GPDM Volume

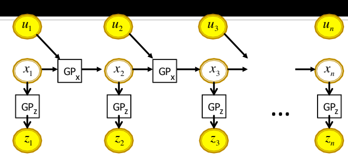
[Wang-Fleet-Hertzmann: PAMI-08]



GPBF-Learning

- Extend GPDMs to
  - incorporate control
  - incorporate sparse labels on latent states
- Steps:
  - Learn GPLVM
  - Extract GP-BayesFilter for tracking

GPBF-Learn



- GPBF-Learn follows same structure as Bayes filter
- Learns latent states and GPs in one optimization
  - GPx for system dynamics
  - GPz for observation model
- Can perform system identification

Probabilistic Framework

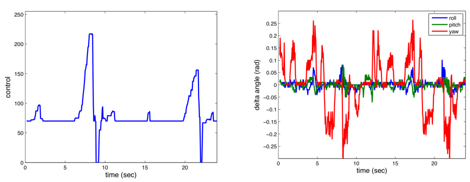
$$p(X,\Theta_Z,\Theta_X \mid Z,U,\hat{X}) \propto$$
$$p(Z \mid X,\Theta_Z)\, p(X \mid \Theta_X,U)\, p(X \mid \hat{X})\, p(\Theta_Z)\, p(\Theta_X)$$

- Unified probabilistic framework incorporating
  - Observation model
  - Dynamics model with controls
  - Weak labels
  - Hyperparameter priors

Evaluation System: Slotcar

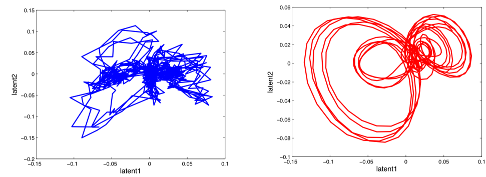
- Track contains banked curves, elevation changes
- Custom IMU with gyros and accelerometers
- Observations very noisy, perceptual aliasing

Data



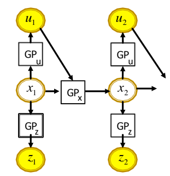
Controls   IMU: angle change rate

Learned 2D Embedding



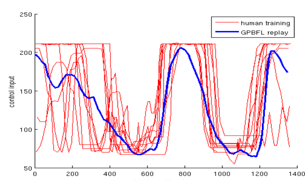
N4SID   GPBF-Learn

Learning by Demonstration with GPBF-Learn



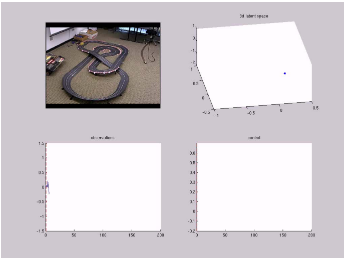
- Learning strategy
  - Learn latent states using GPBF-Learn
  - Learn mapping from state to control
- Replay strategy
  - track state using GP-BayesFilter
  - use control given by control model
- Requires only observations and user demonstrated control

Simple Control Experiment



- Learn control for slot car given human demonstrations
- Replay interpolates human examples
- Fails for more complicated systems

Slotcar Trajectory Replay

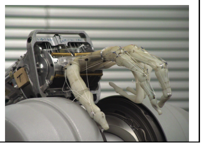


Other Issues

- Heteroscedastic (state dependent) noise
- Non-stationary GPs
- Coupled outputs
- Sparse GPs
  - Online: Decide whether or not to accept new point
  - Remove points
  - Optimize small set of points
- Classification
  - Laplace approximation
  - No closed-form solution, sampling

Summary

- GPs provide flexible modeling framework
- Take data noise and uncertainty due to data sparsity into account
- Combination with parametric models increases accuracy and reduces need for training data
- Computational complexity is a key problem



Some References

- Website: http://www.gaussianprocess.org/
- GP book: http://www.gaussianprocess.org/gpml/
- GPLVM: http://www.cs.man.ac.uk/~neill/gplvm/
- GPDM: http://www.dgp.toronto.edu/~jmwang/gpdm/
- Bishop book: http://research.microsoft.com/en-us/um/people/cmbishop/prml/